

PIC18F6390/6490/8390/8490
数据手册

采用 LCD 驱动器和纳瓦技术的
64/80 引脚闪存单片机

请注意以下有关 Microchip 器件代码保护功能的要点：

- Microchip 的产品均达到 Microchip 数据手册中所述的技术指标。
- Microchip 确信：在正常使用的情况下， Microchip 系列产品是当今市场上同类产品中最安全的产品之一。
- 目前，仍存在着恶意、甚至是非法破坏代码保护功能的行为。就我们所知，所有这些行为都不是以 Microchip 数据手册中规定的操作规范来使用 Microchip 产品的。这样做的人极可能侵犯了知识产权。
- Microchip 愿与那些注重代码完整性的客户合作。
- Microchip 或任何其他半导体厂商均无法保证其代码的安全性。代码保护并不意味着我们保证产品是 “牢不可破”的。

代码保护功能处于持续发展中。 Microchip 承诺将不断改进产品的代码保护功能。任何试图破坏 Microchip 代码保护功能的行为均可视为违反了《数字器件千年版权法案 （Digital Millennium Copyright Act）》。如果这种行为导致他人在未经授权的情况下，能访问您的软件或其他受版权保护的成果，您有权依据该法案提起诉讼，从而制止这种行为。

QUALITY MANAGEMENT SYSTEM
CERTIFIED BY DNV
ISO/TS 16949:2002

Microchip 位于美国亚利桑那州 Chandler 和 Tempe 及位于加利福尼亚州 Mountain View 的全球总部、设计中心和晶圆生产厂均于 2003 年 10 月通过了 ISO/TS-16949:2002 质量体系认证。公司在 PICmicro® 8 位单片机、 KEELOQ® 跳码器件、串行 EEPROM、单片机外设、非易失性存储器和模拟产品方面的质量体系流程均符合 ISO/TS-16949:2002。此外，Microchip 在开发系统的设计和生产方面的质量体系也已通过了 ISO 9001:2000 认证。

MICROCHIP

PIC18F6390/6490/8390/8490

采用 LCD 驱动器和纳瓦技术的 64/80 引脚闪存单片机

LCD 驱动模块的特征：

- 直接驱动 LCD 面板
- 最多 48 个段：可由软件选择
- 可编程 LCD 定时模块：
  - 多个 LCD 定时源
  - 最多 4 个公共时钟：静态、 1/2、 1/3 或 1/4 复用
  - 静态、 1/2 或 1/3 偏置配置
- 处于休眠模式时仍可驱动 LCD 面板

功耗管理模式：

- 运行：CPU 工作，外设打开
- 空闲：CPU 不工作，外设打开
- 休眠：CPU 不工作，外设关闭
- 处于空闲模式时电流可降至 5.8 mA （典型值）
- 处于休眠模式时电流可降至 0.1 mA （典型值）
- Timer1 振荡器：1.8 mA、 32 kHz、 2V
- 看门狗定时器：2.1 mA
- 双速振荡器起振

灵活的振荡器结构：

- 四种晶振模式：
  - LP：频率最高为 200 kHz
  - XT：频率最高为 4 MHz
  - HS：频率最高为 40 MHz
  - HSPLL：频率为 4-10 MHz （内部振荡器频率为 16-40 MHz）
- 4 倍频锁相环 （可用于晶振和内部振荡器）
- 两种外部 RC 模式，频率最高为 4MHz
- 两种外部时钟模式，频率最高为 40MHz
- 内部振荡器电路：
  - 8 个可由用户选择的频率：从 31 kHz 到 8 MHz
  - 当与 PLL 结合使用时可提供较宽的时钟频率范围，从 31 kHz 到 32 MHz
  - 用户可对该电路进行调节以补偿频率漂移
- 辅助振荡器使用 Timer1 （工作频率为 32kHz）
- 故障保护时钟监视器：
  - 当主或辅助时钟发生故障时可使器件安全断电

外设特点：

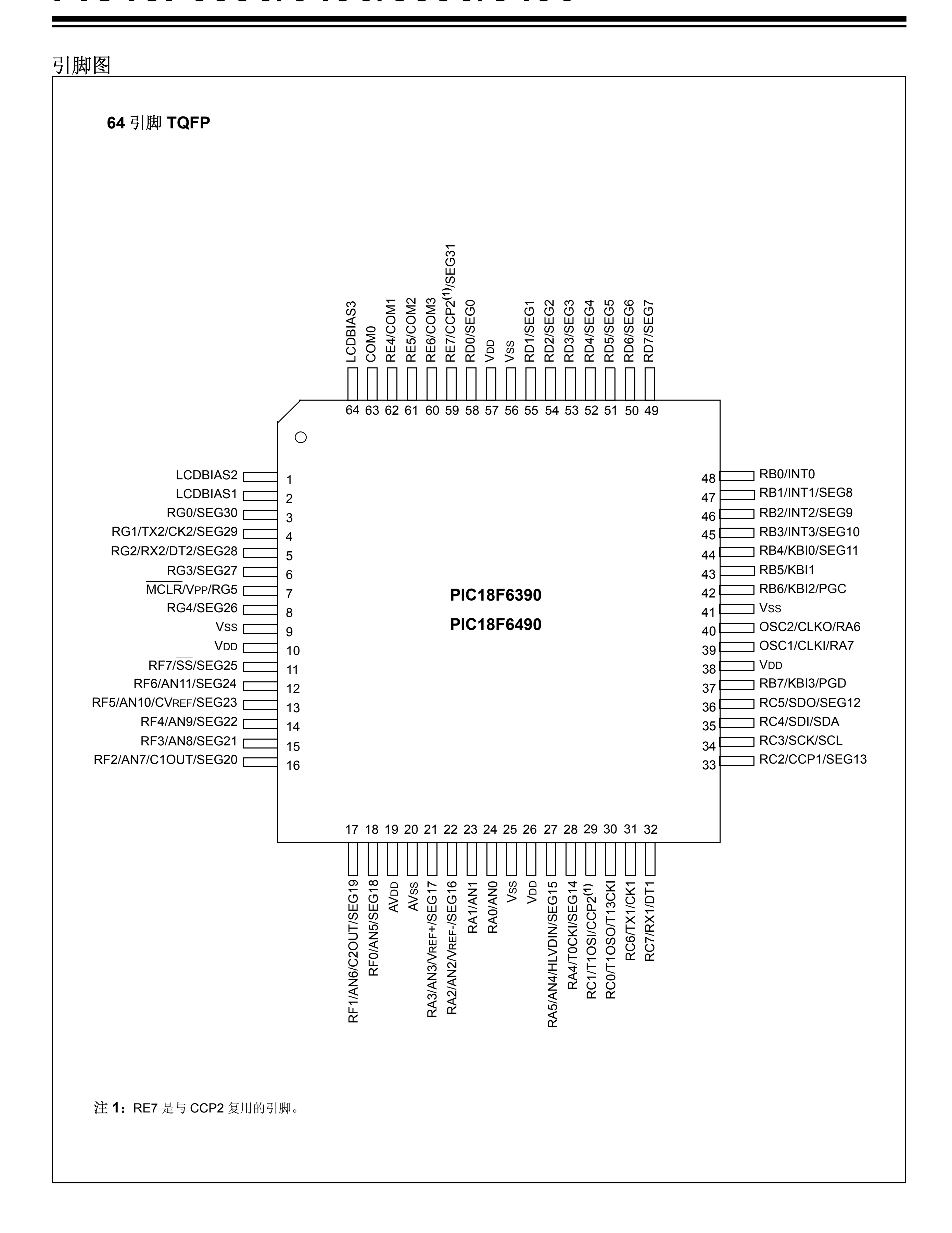
- 高灌 / 拉电流 25 mA/25 mA
- 四个外部中断
- 四个输入电平变化中断
- 四个 8 位 /16 位定时器 / 计数器模块
- 实时时钟 （Real-Time Clock， RTC）软件模块：
  - 可配置的 24 小时时钟、日历、自动的 100 年或 12800 年转换星期日期计算器
  - 使用 Timer1
- 最多两个捕获 / 比较 /PWM （CCP）模块
- 主同步串行口 （Master Synchronous Serial Port， MSSP）模块支持 3 线 SPI™ （总共 4 种模式）和 I²C™ 主从模式
- 可寻址的 USART 模块
  - 支持 RS-485 和 RS-232
- 增强型可寻址 USART 模块：
  - 支持 RS-485、 RS-232 和 LIN 1.2
  - 遇到起始位时自动唤醒
  - 自动波特率检测
- 最多 12 路通道的 10 位模数转换器模块 （A/D）：
  - 自动采样功能
  - 可在休眠状态下进行转换
- 输入复用的双模拟比较器

单片机的特殊功能：

- 优化的 C 编译器架构
  - 为优化重入代码而设计的可选的扩展指令集
- 典型的闪存程序存储器可擦写 1000 次
- 闪存数据保存期：典型值为 100 年
- 中断优先级
- 8 x 8 单周期硬件乘法器
- 扩展的看门狗定时器 （Watchdog Timer， WDT）：
  - 可编程周期从 4 ms 到 132 s
  - 在 VDD 电压和温度变化 ± 2% 范围内保持稳定
- 通过两个引脚进行在线串行编程 （In-Circuit Serial Programming™， ICSP™）
- 通过两个引脚进行在线调试 （In-Circuit Debug， ICD）
- 宽工作电压范围：2.0V 到 5.5V

器件	程序存储器		数据存储器	I/O	LCD （像素）	10 位 A/D (通道)	CCP (PWM)	MSSP		EUSART/AUSART	比较器	8/16 位定时器
	闪存 （字节）	单字指令	SRAM （字节）					SPI	主控 I²C™			
PIC18F6390	8K	4096	768	50	128	12	2	有	有	1/1	2	1/3
PIC18F6490	16K	8192	768	50	128	12	2	有	有	1/1	2	1/3
PIC18F8390	8K	4096	768	66	192	12	2	有	有	1/1	2	1/3
PIC18F8490	16K	8192	768	66	192	12	2	有	有	1/1	2	1/3

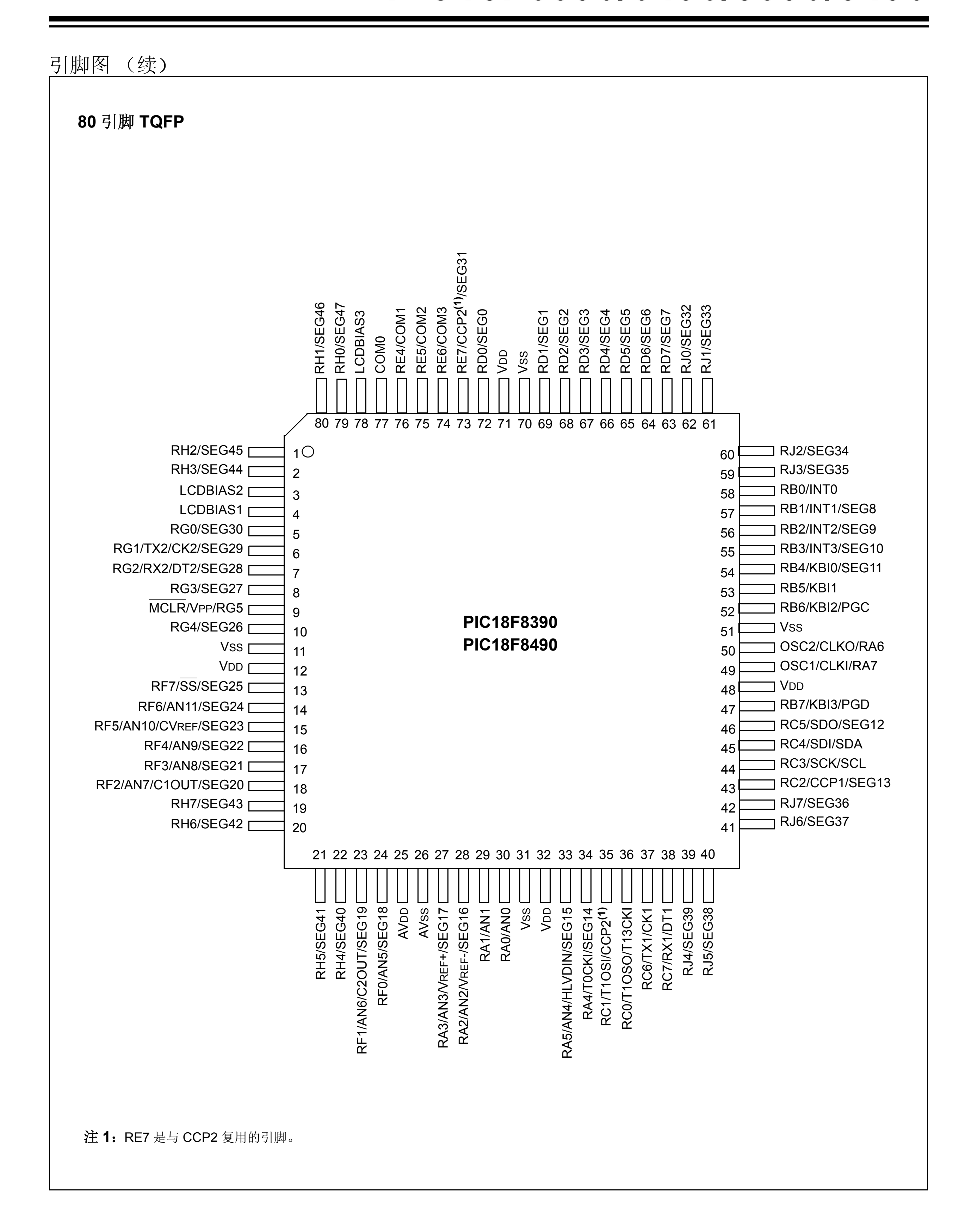
引脚图



注 1：RE7 是与 CCP2 复用的引脚。

 初稿

引脚图 （续）



注 1： RE7 是与 CCP2 复用的引脚。

目录

 初稿

致 客 户

我们旨在提供最佳文档供客户正确使用 Microchip 产品。为此，我们将不断改进出版物的内容和质量，使之更好地满足您的要求。出版物的质量将随新文档及更新版本的推出而得到提升。

如果您对本出版物有任何问题和建议，请通过电子邮件联系我公司 TRC 经理，电子邮件地址为 **CTRC@microchip.com**，或将本数据手册后附的 **《读者反馈表》**传真到 86-21-5407 5066。我们期待您的反馈。

最新数据手册

欲获得本数据手册的最新版本，请查询我公司的网站：

http://www.microchip.com

查看数据手册中任意一页下边角处的文献编号即可确定其版本。文献编号中数字串后的字母是版本号，例如：DS30000A是DS30000 的 A 版本。

勘误表

现有器件可能带有一份勘误表，描述了实际运行与数据手册中记载内容之间存在的细微差异以及建议的变通方法。一旦我们了解到器件 / 文档存在某些差异时，就会发布勘误表。勘误表上将注明其所适用的硅片版本和文件版本。

欲了解某一器件是否存在勘误表，请通过以下方式之一查询：

- Microchip 网站 http://www.microchip.com
- 当地 Microchip 销售办事处 （见最后一页）

在联络销售办事处时，请说明您所使用的器件型号、硅片版本和数据手册版本 （包括文献编号）。

客户通知系统

欲及时获知 Microchip 产品的最新信息，请到我公司网站 **www.microchip.com** 上注册。

注：

1.0 器件概述

此文档包含针对以下器件的信息：

- PIC18F6390
- PIC18F6490
- PIC18F8390
- PIC18F8490

该系列具备所有 PIC18 单片机固有的优点——即，实惠的价格和出色的计算性能。除此之外，PIC18F6390/6490/8390/8490 系列还引进了增强型设计，使得该系列单片机成为许多高性能和节能应用的明智选择。

1.1 新的内核功能

1.1.1 纳瓦技术

PIC18F6390/6490/8390/8490 系列的所有器件具有一系列能在工作时显著降低功耗的功能。主要包含以下几项：

- **备用运行模式：**通过将 Timer1 或内部振荡器模块作为单片机时钟源，可使代码执行时的功耗降低大约 90%。
- **多种空闲模式：**单片机还可工作在其 CPU 内核禁止而外设仍然运行的情况下。处于这些状态时，功耗能降得更低，只有正常工作时的 4%。
- **动态模式切换：**在器件工作期间可由用户代码调用该功耗管理模式，允许用户将节能的理念融入到他们的应用软件设计中。
- **较低的关键模块功耗：**Timer1 和看门狗定时器模块的能耗需求可降低最多至 80%，两者的典型值分别为 1.1 µA 和 2.1 µA。

1.1.2 多个振荡器选项和特点

PIC18F6390/6490/8390/8490 系列的所有器件可提供 9 个不同的振荡器选项，使用户在开发应用硬件时有很大的选择范围。这些选项包括：

- 四个晶振模式，使用晶振或陶瓷谐振器。
- 两个外部时钟模式，提供使用两个引脚 （振荡器输入引脚和四分频时钟输出引脚）或一个引脚 （振荡器输入引脚，四分频时钟输出引脚重新分配为通用 I/O 引脚）的选项。
- 两个外部 RC 振荡器模式，具有与外部时钟模式相同的引脚选项。
- 一个内部振荡器模块，它提供一个 8 MHz 的时钟源 （精度为± 2%）和一个内部 RC 时钟源 （振荡频率大约为 31 kHz，温度和 V_{DD} 变化时频率保持稳定），以及一个用户可选择的包括 6 种时钟频率的范围 （从 125 kHz 到 4 MHz），因此共有 8 种时钟频率可供选择。此选项可以空出两个振荡器引脚作为额外的通用 I/O 引脚。
- 一个锁相环 （Phase Lock Loop， PLL）倍频器，可在高速晶振和内部振荡器模式下使用，这两种模式可使时钟 速度最高达到 40 MHz。PLL 和内部振荡器配合使用，可以向用户提供频率范围从 31 kHz 到 32 MHz 的时钟速度以供选择，而且不需要使用外部晶振或时钟电路。

除了可被用作时钟源外，内部振荡器模块还提供了一个稳定的参考源，增加了以下功能以使器件更安全地工作：

- **故障保护时钟监视器：**该部件不断监视主时钟源，将其与内部振荡器提供的参考信号作比较。如果主时钟发生了故障，单片机会将时钟源切换到内部振荡器模块，使器件可继续低速工作或安全地关机。
- **双速启动：**该功能允许在上电复位或从休眠模式唤醒时将内部振荡器用作时钟源，直到主时钟源可正常工作为止。

1.2 其他特殊功能

- **存储器耐久性：**程序存储器擦写周期额定值大约为 1000 次。如果不刷新，数据保存期保守地估计在 100 年以上。
- **扩展的指令集：**PIC18F6390/6490/8390/8490 系列在 PIC18 指令集的基础上进行了可选择的扩展，添加了 8 个新指令和索引寻址模式。此扩展可以使用一个器件配置选项使能，它是为优化重入应用程序代码而特别设计的，这些代码原来是使用高级语言（如 C 语言）开发的。
- **增强型可寻址 USART：**此串行通信模块可进行标准的 RS-232 通信并支持 LIN 总线协议。其他增强的功能包括自动波特率检测和精度更高的 16 位波特率发生器。当单片机使用内部振荡器模块时，EUSART 为与外界对话的应用程序提供稳定的通信方式，而无需使用外部晶振也无需额外的功耗。
- **10 位 A/D 转换器：**该模块具备可编程采集时间，从而不必在选择通道和启动转换之间等待一个采样周期，因而减少了代码开销。
- **扩展的看门狗定时器（WDT）：**该增强的看门狗定时器添加了 16 位预分频器，其定时范围可从 4 ms 到大于 10 分钟，并能在工作电压和温度发生变化时保持稳定。

1.3 系列中各产品的详细说明

PIC18F6390/6490/8390/8490 系列器件具有 64 引脚（PIC18F6X90）和 80 引脚（PIC18F8X90）两种封装形式。图 1-1 和图 1-2 分别为这两类器件的框图。

这两类器件在以下三个方面存在差异：

1. I/O 口：64 引脚器件上有 7 个双向端口，而 80 引脚器件上有 9 个双向端口。
2. LCD 像素：64 引脚器件可驱动 128（32 个 SEG x 4 个 COM）像素，而 80 引脚器件可驱动 192（48 个 SEG x 4 个 COM）像素。
3. 闪存程序存储器：PIC18FX390 器件为 8K 字节，而 PIC18FX490 为 16K 字节。

本系列器件的其他功能都是相同的。表 1-1 汇总了这些功能。

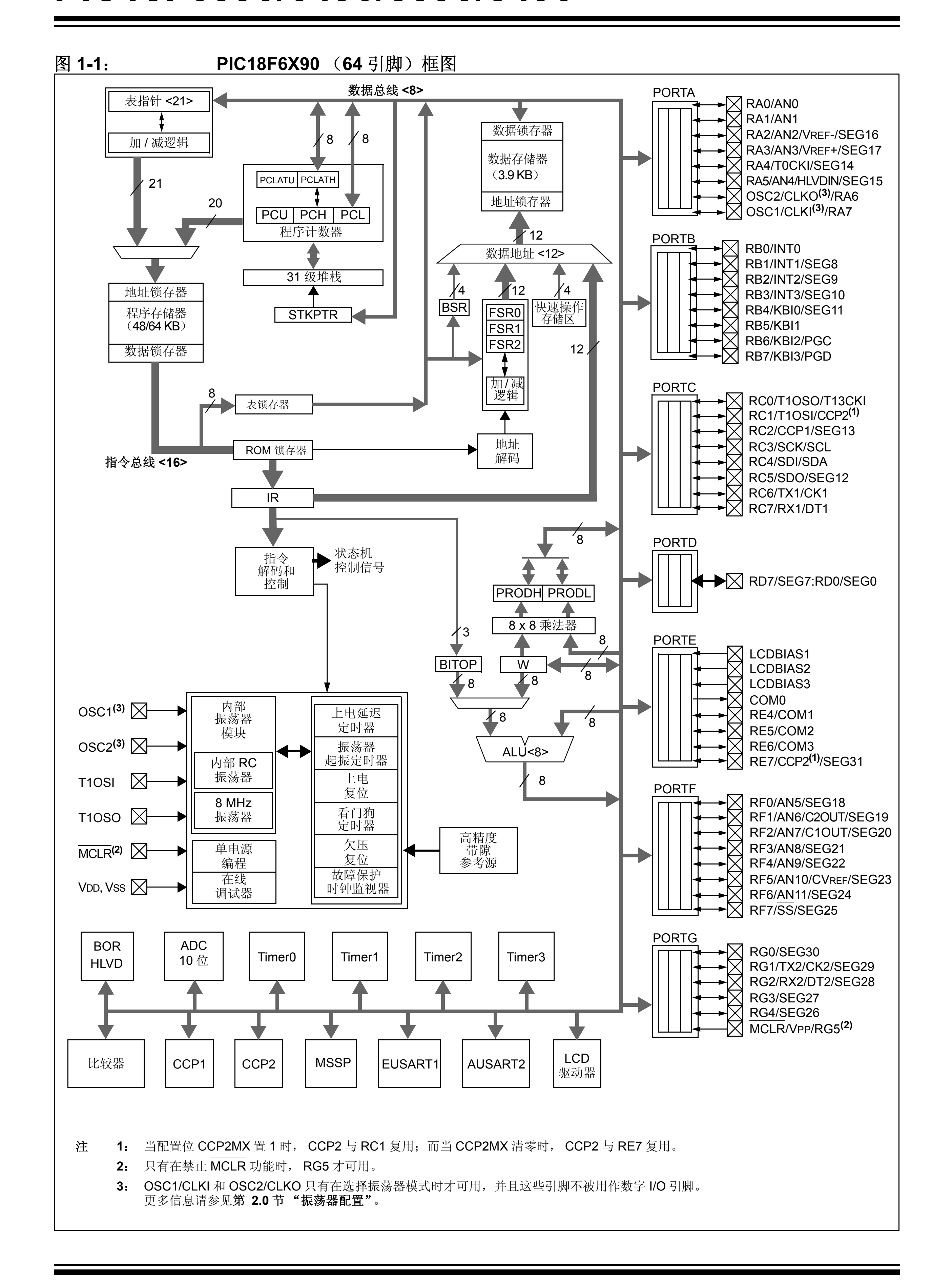
表 1-2 和表 1-3 给出了本系列中所有器件的引脚说明。

和所有的 Microchip PIC18 器件一样，PIC18F6390/6490/8390/8490 系列中也有标准器件和低压器件。器件编号中标有字母 “F” 的是带有闪存存储器的标准器件（如 PIC18F6390），其工作电压 V_{DD} 范围为 4.2V 到 5.5V。编号中标有 “LF” 的低压器件（如 PIC18LF6490）可工作在扩展的 V_{DD} 范围（2.0V 到 5.5V）中。

表 **1-1**： 器件特性

特性	PIC18F6390	PIC18F6490	PIC18F8390	PIC18F8490
工作频率	DC–40 MHz	DC–40 MHz	DC–40 MHz	DC–40 MHz
程序存储器 （字节）	8K	16K	8K	16K
程序存储器 （指令）	4096	8192	4096	8192
数据存储器 （字节）	768	768	768	768
中断源	22	22	22	22
I/O 口	端口 A、 B、 C、 D、 E、 F 和 G	端口 A、 B、 C、 D、 E、 F 和 G	端口 A、 B、 C、 D、 E、 F、 G、 H 和 J	端口 A、 B、 C、 D、 E、 F、 G、 H 和 J
LCD 驱动器可以驱动的像素数量	128 （32 个 SEG x 4 个 COM）	128 （32 个 SEG x 4 个 COM）	192 （48 个 SEG x 4 个 COM）	192 （48 个 SEG x 4 个 COM）
定时器	4	4	4	4
捕捉 / 比较 /PWM 模块	2	2	2	2
串行通信	MSSP、AUSART 增强型 USART	MSSP、AUSART 增强型 USART	MSSP、AUSART 增强型 USART	MSSP、 AUSART 增强型 USART
10 位模数转换模块	12 路输入通道	12 路输入通道	12 路输入通道	12 路输入通道
复位 （和延迟）	POR、 BOR、 RESET 指令， 堆栈满、 堆栈下溢 （PWRT 和 OST）、 $\overline{\text{MCLR}}$ （可选） 和 WDT	POR、 BOR、 RESET 指令， 堆栈满、 堆栈下溢 （PWRT 和 OST）、 $\overline{\text{MCLR}}$ （可选） 和 WDT	POR、 BOR、 RESET 指令， 堆栈满、 堆栈下溢 （PWRT 和 OST）、 $\overline{\text{MCLR}}$ （可选） 和 WDT	POR、 BOR、 RESET 指令， 堆栈满、 堆栈下溢 （PWRT 和 OST）、 $\overline{\text{MCLR}}$ （可选） 和 WDT
可编程低压检测	有	有	有	有
可编程欠压复位	有	有	有	有
指令集	75 条指令； 启用了扩展指令集后总共为 83 条指令	75 条指令； 启用了扩展指令集后总共为 83 条指令	75 条指令； 启用了扩展指令集后总共为 83 条指令	75 条指令； 启用了扩展指令集后总共为 83 条指令
封装	64 引脚 TQFP	64 引脚 TQFP	80 引脚 TQFP	80 引脚 TQFP

图 1-1： PIC18F6X90 （64 引脚）框图

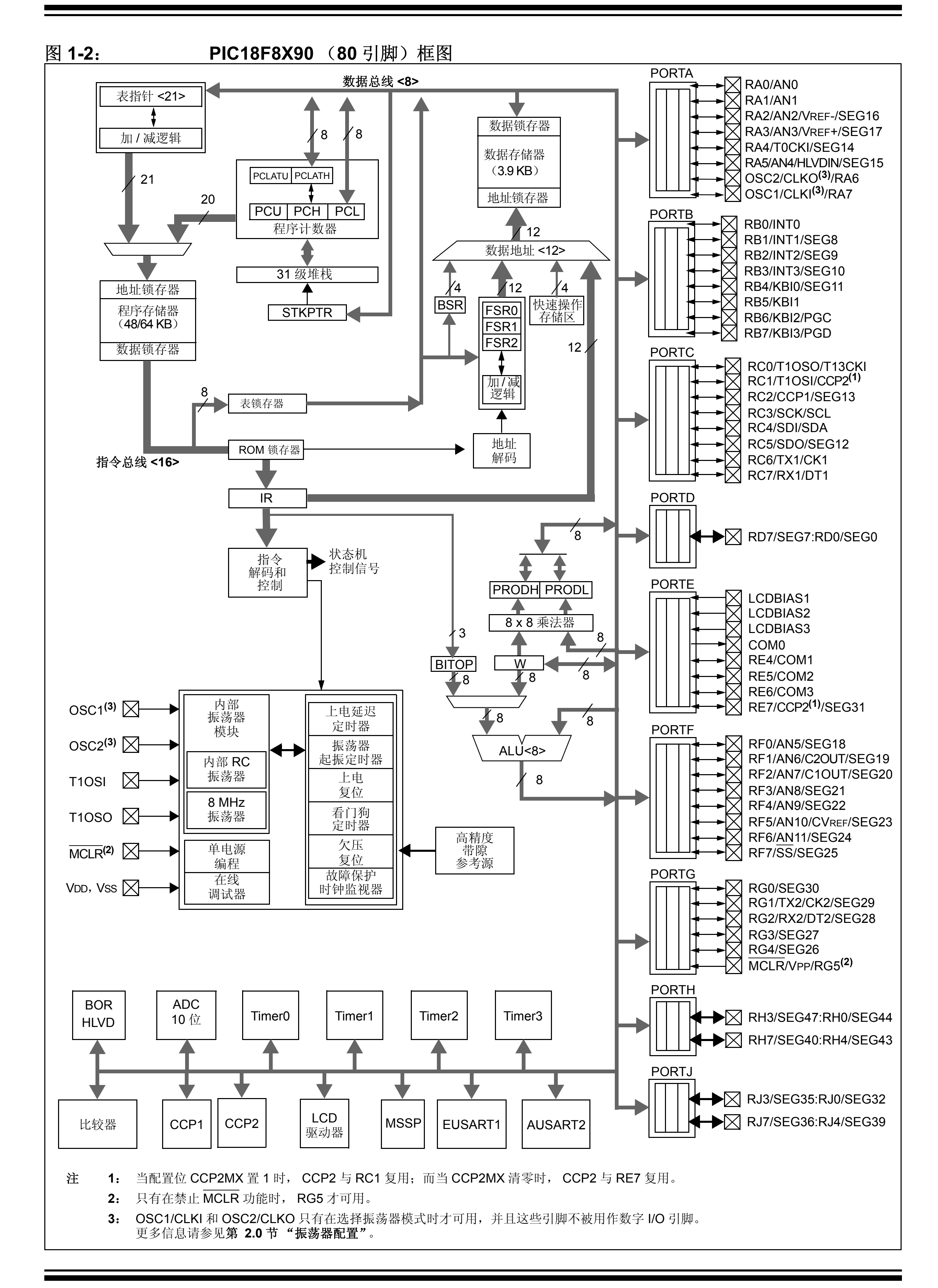


注 1： 当配置位 CCP2MX 置 1 时， CCP2 与 RC1 复用；而当 CCP2MX 清零时， CCP2 与 RE7 复用。

2： 只有在禁止 MCLR 功能时， RG5 才可用。

3： OSC1/CLKI 和 OSC2/CLKO 只有在选择振荡器模式时才可用，并且这些引脚不被用作数字 I/O 引脚。更多信息请参见**第 2.0 节 “振荡器配置”**。

图 1-2: PIC18F8X90 （80 引脚）框图



注 1： 当配置位 CCP2MX 置 1 时， CCP2 与 RC1 复用；而当 CCP2MX 清零时， CCP2 与 RE7 复用。

2： 只有在禁止 MCLR 功能时， RG5 才可用。

3： OSC1/CLKI 和 OSC2/CLKO 只有在选择振荡器模式时才可用，并且这些引脚不被用作数字 I/O 引脚。更多信息请参见**第 2.0 节 “振荡器配置”**。

表 **1-2**： **PIC18F6X90 I/O** 引脚说明

引脚名称	引脚号 TQFP	引脚类型	缓冲器类型	说明
$\overline{MCLR}$/VPP/RG5	7			主清零 （输入）或编程电压 （输入）。
MCLR		I	ST	主清零 （复位）输入。此引脚为低电平时，器件复位。
VPP		P		编程电压输入。
RG5		I	ST	数字输入。
OSC1/CLKI/RA7	39			振荡器晶振或外部时钟输入。
OSC1		I	ST	振荡器晶振输入或外部时钟源输入。在 RC 模式下带 ST 缓冲器，否则带 CMOS 缓冲器。
CLKI		I	CMOS	外部时钟源输入。总是与OSC1 引脚功能复用 （见相关的 OSC1/CLKI 和 OSC2/CLKO 引脚信息）。
RA7		I/O	TTL	通用 I/O 引脚。
OSC2/CLKO/RA6	40			振荡器晶振或时钟输出。
OSC2		O	—	振荡器晶振输出。在晶振模式下，该引脚与晶振或谐振器相连。
CLKO		O	—	在 RC 模式下， OSC2 引脚输出 CLKO 振荡信号，该信号是 OSC1 引脚上振荡信号的 4 分频，该频率等于指令周期的倒数。
RA6		I/O	TTL	通用 I/O 引脚。

图注： TTL = TTL 兼容输入　　CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器输入　　Analog = 模拟输入
I = 输入　　O = 输出
P = 电源　　OD = 漏极开路 （没有 P 型二极管接到 VDD）

注 **1**： 当配置位 CCP2MX 置 1 时，对 CCP2 进行默认分配。
2： 当配置位 CCP2MX 清零时，对 CCP2 进行其他分配。

表 **1-2：** **PIC18F6X90 I/O** 引脚说明 （续）

引脚名称	引脚号 TQFP	引脚类型	缓冲器类型	说明
				PORTA 是双向 I/O 口。
RA0/AN0	24			
RA0		I/O	TTL	数字 I/O。
AN0		I	模拟	模拟输入 0。
RA1/AN1	23			
RA1		I/O	TTL	数字 I/O。
AN1		I	模拟	模拟输入 1。
RA2/AN2/VREF-/SEG16	22			
RA2		I/O	TTL	数字 I/O。
AN2		I	模拟	模拟输入 2。
VREF-		I	模拟	A/D 参考电压 （低电平）输入。
SEG16		O	模拟	LCD 的 SEG16 输出。
RA3/AN3/VREF+/SEG17	21			
RA3		I/O	TTL	数字 I/O。
AN3		I	模拟	模拟输入 3。
VREF+		I	模拟	A/D 参考电压 （高电平）输入。
SEG17		O	模拟	LCD 的 SEG17 输出。
RA4/T0CKI/SEG14	28			
RA4		I/O	ST/OD	数字 I/O。在配置为输出时漏极开路。
T0CKI		I	ST	Timer0 外部时钟输入。
SEG14		O	模拟	LCD 的 SEG14 输出。
RA5/AN4/HLVDIN/SEG15	27			
RA5		I/O	TTL	数字 I/O。
AN4		I	模拟	模拟输入 4。
HLVDIN		I	模拟	低压检测输入。
SEG15		O	模拟	LCD 的 SEG15 输出。
RA6				请参见 OSC2/CLKO/RA6 引脚信息。
RA7				请参见 OSC1/CLKI/RA7 引脚信息。

图注： TTL = TTL 兼容输入　　CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器输入　　Analog = 模拟输入
I = 输入　　O = 输出
P = 电源　　OD = 漏极开路 （没有 P 型二极管接到 VDD）

注 **1：** 当配置位 CCP2MX 置 1 时，对 CCP2 进行默认分配。
2： 当配置位 CCP2MX 清零时，对 CCP2 进行其他分配。

表 **1-2：** **PIC18F6X90 I/O** 引脚说明 （续）

引脚名称	引脚号 TQFP	引脚类型	缓冲器类型	说明
				PORTB 是双向 I/O 口。PORTB 在所有输入端都可软件编程为内部弱上拉。
RB0/INT0	48			
RB0		I/O	TTL	数字 I/O。
INT0		I	ST	外部中断 0。
RB1/INT1/SEG8	47			
RB1		I/O	TTL	数字 I/O。
INT1		I	ST	外部中断 1。
SEG8		O	模拟	LCD 的 SEG8 输出。
RB2/INT2/SEG9	46			
RB2		I/O	TTL	数字 I/O。
INT2		I	ST	外部中断 2。
SEG9		O	模拟	LCD 的 SEG9 输出。
RB3/INT3/SEG10	45			
RB3		I/O	TTL	数字 I/O。
INT1		I	ST	外部中断 3。
SEG10		O	模拟	LCD 的 SEG10 输出。
RB4/KBI0/SEG11	44			
RB4		I/O	TTL	数字 I/O。
KBI0		I	TTL	电平变化中断引脚。
SEG11		O	模拟	LCD 的 SEG11 输出。
RB5/KBI1	43			
RB5		I/O	TTL	数字 I/O。
KBI1		I	TTL	电平变化中断引脚。
RB6/KBI2/PGC	42			
RB6		I/O	TTL	数字 I/O。
KBI2		I	TTL	电平变化中断引脚。
PGC		I/O	ST	在线调试器和 ICSP™ 编程时钟引脚。
RB7/KBI3/PGD	37			
RB7		I/O	TTL	数字 I/O。
KBI3		I	TTL	电平变化中断引脚。
PGD		I/O	ST	在线调试器和 ICSP 编程数据引脚。

图注： TTL = TTL 兼容输入 CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器输入 Analog = 模拟输入
I = 输入 O = 输出
P = 电源 OD = 漏极开路 （没有 P 型二极管接到 VDD）

注 **1：** 当配置位 CCP2MX 置 1 时，对 CCP2 进行默认分配。
2： 当配置位 CCP2MX 清零时，对 CCP2 进行其他分配。

表 1-2： PIC18F6X90 I/O 引脚说明 （续）

引脚名称	引脚号 TQFP	引脚类型	缓冲器类型	说明
				PORTC 是双向 I/O 口。
RC0/T1OSO/T13CKI	30			
RC0		I/O	ST	数字 I/O。
T1OSO		O	—	Timer1 振荡器输出。
T13CKI		I	ST	Timer1/Timer3 外部时钟输入。
RC1/T1OSI/CCP2	29			
RC1		I/O	ST	数字 I/O。
T1OSI		I	CMOS	Timer1 振荡器输入。
CCP2(1)		I/O	ST	Capture2 输入 /Compare2 输出 /PWM2 输出
RC2/CCP1/SEG13	33			
RC2		I/O	ST	数字 I/O。
CCP1		I/O	ST	Capture1 输入 /Compare1 输出 /PWM1 输出。
SEG13		O	模拟	LCD 的 SEG13 输出。
RC3/SCK/SCL	34			
RC3		I/O	ST	数字 I/O。
SCK		I/O	ST	SPI™ 模式的同步串行时钟输入 / 输出。
SCL		I/O	ST	I^2C™ 模式的同步串行时钟输入 / 输出。
RC4/SDI/SDA	35			
RC4		I/O	ST	数字 I/O。
SDI		I	ST	SPI 数据输入。
SDA		I/O	ST	I^2C 数据 I/O。
RC5/SDO/SEG12	36			
RC5		I/O	ST	数字 I/O。
SDO		O	—	SPI 数据输出。
SEG12		O	模拟	LCD 的 SEG12 输出。
RC6/TX1/CK1	31			
RC6		I/O	ST	数字 I/O。
TX1		O	—	EUSART1 异步发送。
CK1		I/O	ST	EUSART1 同步时钟 （见 RX1/DT1 引脚信息）。
RC7/RX1/DT1	32			
RC7		I/O	ST	数字 I/O。
RX1		I	ST	EUSART1 异步接收。
DT1		I/O	ST	EUSART1 同步数据 （见 TX1/CK1 引脚信息）。

图注： TTL = TTL 兼容输入　　CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器输入　　Analog = 模拟输入
I = 输入　　O = 输出
P = 电源　　OD = 漏极开路 （没有 P 型二极管接到 VDD）

注 1： 当配置位 CCP2MX 置 1 时，对 CCP2 进行默认分配。
2： 当配置位 CCP2MX 清零时，对 CCP2 进行其他分配。

表 **1-2：** **PIC18F6X90 I/O** 引脚说明 （续）

引脚名称	引脚号 TQFP	引脚类型	缓冲器类型	说明
				PORTD 是双向 I/O 口。
RD0/SEG0	58			
RD0		I/O	ST	数字 I/O。
SEG0		O	模拟	LCD 的 SEG0 输出。
RD1/SEG1	55			
RD1		I/O	ST	数字 I/O。
SEG1		O	模拟	LCD 的 SEG1 输出。
RD2/SEG2	54			
RD2		I/O	ST	数字 I/O。
SEG2		O	模拟	LCD 的 SEG2 输出。
RD3/SEG3	53			
RD3		I/O	ST	数字 I/O。
SEG3		O	模拟	LCD 的 SEG3 输出。
RD4/SEG4	52			
RD4		I/O	ST	数字 I/O。
SEG4		O	模拟	LCD 的 SEG4 输出。
RD5/SEG5	51			
RD5		I/O	ST	数字 I/O。
SEG5		O	模拟	LCD 的 SEG5 输出。
RD6/SEG6	50			
RD6		I/O	ST	数字 I/O。
SEG6		O	模拟	LCD 的 SEG6 输出。
RD7/SEG7	49			
RD7		I/O	ST	数字 I/O。
SEG7		O	模拟	LCD 的 SEG7 输出。

图注： TTL = TTL 兼容输入　　CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器输入　　Analog = 模拟输入
I = 输入　　O = 输出
P = 电源　　OD = 漏极开路 （没有 P 型二极管接到 V_{DD}）

注 **1：** 当配置位 CCP2MX 置 1 时，对 CCP2 进行默认分配。
2： 当配置位 CCP2MX 清零时，对 CCP2 进行其他分配。

表 **1-2**： **PIC18F6X90 I/O** 引脚说明 （续）

引脚名称	引脚号 TQFP	引脚类型	缓冲器类型	说明
				PORTE 是双向 I/O 口。
LCDBIAS1	2			
LCDBIAS1		I	模拟	LCD 的 BIAS1 输入。
LCDBIAS2	1			
LCDBIAS2		I	模拟	LCD 的 BIAS2 输入。
LCDBIAS3	64			
LCDBIAS3		I	模拟	LCD 的 BIAS3 输入。
COM0	63			
COM0		O	模拟	LCD 的 COM0 输出。
RE4/COM1	62			
RE4		I/O	ST	数字 I/O。
COM1		O	模拟	LCD 的 COM1 输出。
RE5/COM2	61			
RE5		I/O	ST	数字 I/O。
COM2		O	模拟	LCD 的 COM2 输出。
RE6/COM3	60			
RE6		I/O	ST	数字 I/O。
COM3		O	模拟	LCD 的 COM3 输出。
RE7/CCP2/SEG31	59			
RE7		I/O	ST	数字 I/O。
CCP2[(2)]		I/O	ST	Capture 2 输入 /Compare 2 输出 /PWM 2 输出。
SEG31		O	模拟	LCD 的 SEG31 输出。

图注： TTL = TTL 兼容输入　　CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器输入　　Analog = 模拟输入
I = 输入　　O = 输出
P = 电源　　OD = 漏极开路 （没有 P 型二极管接到 V_{DD}）

注 **1**： 当配置位 CCP2MX 置 1 时，对 CCP2 进行默认分配。
2： 当配置位 CCP2MX 清零时，对 CCP2 进行其他分配。

表 **1-2**： **PIC18F6X90 I/O** 引脚说明 （续）

引脚名称	引脚号 TQFP	引脚类型	缓冲器类型	说明
				PORTF 是双向 I/O 口。
RF0/AN5/SEG18	18			
RF0		I/O	ST	数字 I/O。
AN5		I	模拟	模拟输入 5。
SEG18		O	模拟	LCD 的 SEG18 输出。
RF1/AN6/C2OUT/SEG19	17			
RF1		I/O	ST	数字 I/O。
AN6		I	模拟	模拟输入 6。
C2OUT		O	—	Comparator 2 输出。
SEG19		O	模拟	LCD 的 SEG19 输出。
RF2/AN7/C1OUT/SEG20	16			
RF2		I/O	ST	数字 I/O。
AN7		I	模拟	模拟输入 7。
C1OUT		O	—	Comparator 1 输出。
SEG20		O	模拟	LCD 的 SEG20 输出。
RF3/AN8/SEG21	15			
RF3		I/O	ST	数字 I/O。
AN8		I	模拟	模拟输入 8。
SEG21		O	模拟	LCD 的 SEG21 输出。
RF4/AN9/SEG22	14			
RF4		I/O	ST	数字 I/O。
AN9		I	模拟	模拟输入 9。
SEG22		O	模拟	LCD 的 SEG22 输出。
RF5/AN10/CVREF/SEG23	13			
RF5		I/O	ST	数字 I/O。
AN10		I	模拟	模拟输入 10。
CVREF		O	模拟	比较器参考电压输出。
SEG23		O	模拟	LCD 的 SEG23 输出。
RF6/AN11/SEG24	12			
RF6		I/O	ST	数字 I/O。
AN11		I	模拟	模拟输入 11。
SEG24		O	模拟	LCD 的 SEG24 输出。
RF7/$\overline{SS}$/SEG25	11			
RF7		I/O	ST	数字 I/O。
$\overline{SS}$		I	TTL	SPI™ 从动选择输入。
SEG25		O	模拟	LCD 的 SEG25 输出。

图注： TTL = TTL 兼容输入　　CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器输入　　Analog = 模拟输入
I = 输入　　O = 输出
P = 电源　　OD = 漏极开路 （没有 P 型二极管接到 VDD）

注 **1**： 当配置位 CCP2MX 置 1 时，对 CCP2 进行默认分配。
2： 当配置位 CCP2MX 清零时，对 CCP2 进行其他分配。

表 **1-2**： **PIC18F6X90 I/O** 引脚说明 （续）

引脚名称	引脚号 TQFP	引脚类型	缓冲器类型	说明
				PORTG 是双向 I/O 口。
RG0/SEG30	3			
RG0		I/O	ST	数字 I/O。
SEG30		O	模拟	LCD 的 SEG30 输出。
RG1/TX2/CK2/SEG29	4			
RG1		I/O	ST	数字 I/O。
TX2		O	—	AUSART2 异步发送。
CK2		I/O	ST	AUSART2 同步时钟 （见 RX2/DT2 引脚信息）。
SEG29		O	模拟	LCD 的 SEG29 输出。
RG2/RX2/DT2/SEG28	5			
RG2		I/O	ST	数字 I/O。
RX2		I	ST	AUSART2 异步接收。
DT2		I/O	ST	AUSART2 同步数据 （见 TX2/CK2 引脚信息）。
SEG28		O	模拟	LCD 的 SEG28 输出。
RG3/SEG27	6			
RG3		I/O	ST	数字 I/O。
SEG27		O	模拟	LCD 的 SEG27 输出。
RG4/SEG26	8			
RG4		I/O	ST	数字 I/O。
SEG26		O	模拟	LCD 的 SEG26 输出。
RG5				请参见 $\overline{\text{MCLR}}$/VPP/RG5 引脚信息。
VSS	9, 25, 41, 56	P	—	逻辑电路和 I/O 引脚的参考地。
VDD	10, 26, 38, 57	P	—	逻辑电路和 I/O 引脚的正向电源。
AVSS	20	P	—	模拟模块的参考地。
AVDD	19	P	—	模拟模块的正向电源。

图注：
TTL = TTL 兼容输入
CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器输入
Analog = 模拟输入
I = 输入
O = 输出
P = 电源
OD = 漏极开路 （没有 P 型二极管接到 VDD）

注 **1**： 当配置位 CCP2MX 置 1 时，对 CCP2 进行默认分配。
2： 当配置位 CCP2MX 清零时，对 CCP2 进行其他分配。

表 1-3： PIC18F8X90 I/O 引脚说明

引脚名称	引脚号 TQFP	引脚类型	缓冲器类型	说明
$\overline{MCLR}$/VPP/RG5	9			主清零 （输入）或编程电压 （输入）。
MCLR		I	ST	主清零 （复位）输入。此引脚为低电平时，器件复位。
VPP		P		编程电压输入。
RG5		I	ST	数字输入。
OSC1/CLKI/RA7	49			振荡器晶振或外部时钟输入。
OSC1		I	ST	振荡器晶振输入或外部时钟源输入。在 RC 模式配置时带 ST 缓冲器，否则带 CMOS 缓冲器。
CLKI		I	CMOS	外部时钟源输入。总是与 OSC1 引脚功能复用 （见 OSC1/CLKI，OSC2/CLKO 引脚信息）。
RA7		I/O	TTL	通用 I/O 引脚。
OSC2/CLKO/RA6	50			振荡器晶振或时钟输出。
OSC2		O	—	振荡器晶振输出。在晶振模式下，该引脚与晶振或谐振器相连。
CLKO		O	—	在 RC 模式下， OSC2 引脚输出 CLKO 振荡信号，该信号是 OSC1 引脚上振荡信号的 4 分频，该频率等于指令周期的倒数。
RA6		I/O	TTL	通用 I/O 引脚。

图注： TTL = TTL 兼容输入 CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器输入 Analog = 模拟输入
I = 输入 O = 输出
P = 电源 OD = 漏极开路 （没有 P 型二极管接到 VDD）

注 1： 当配置位 CCP2MX 置 1 时，对 CCP2 进行默认分配。
2： 当配置位 CCP2MX 清零时，对 CCP2 进行其他分配。

表 **1-3：　　PIC18F8X90 I/O** 引脚说明　（续）

引脚名称	引脚号 TQFP	引脚类型	缓冲器类型	说明
				PORTA 是双向 I/O 口。
RA0/AN0	30			
RA0		I/O	TTL	数字 I/O。
AN0		I	模拟	模拟输入 0。
RA1/AN1	29			
RA1		I/O	TTL	数字 I/O。
AN1		I	模拟	模拟输入 1。
RA2/AN2/VREF-/SEG16	28			
RA2		I/O	TTL	数字 I/O。
AN2		I	模拟	模拟输入 2。
VREF-		I	模拟	A/D 参考电压 （低电平）输入。
SEG16		O	模拟	LCD 的 SEG16 输出。
RA3/AN3/VREF+/SEG17	27			
RA3		I/O	TTL	数字 I/O。
AN3		I	模拟	模拟输入 3。
VREF+		I	模拟	A/D 参考电压 （高电平）输入。
SEG17		O	模拟	LCD 的 SEG17 输出。
RA4/T0CKI/SEG14	34			
RA4		I/O	ST/OD	数字 I/O。在配置为输出时漏极开路。
T0CKI		I	ST	Timer0 外部时钟源输入。
SEG14		O	模拟	LCD 的 SEG14 输出。
RA5/AN4/HLVDIN/SEG15	33			
RA5		I/O	TTL	数字 I/O。
AN4		I	模拟	模拟输入 4。
HLVDIN		I	模拟	低压检测输入。
SEG15		O	模拟	LCD 的 SEG15 输出。
RA6				请参见 OSC2/CLKO/RA6 引脚信息。
RA7				请参见 OSC1/CLKI/RA7 引脚信息。

图注：
TTL = TTL 兼容输入　　CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器输入　　Analog = 模拟输入
I = 输入　　O = 输出
P = 电源　　OD = 漏极开路 （没有 P 型二极管接到 VDD）

注 **1：** 当配置位 CCP2MX 置 1 时，对 CCP2 进行默认分配。
2： 当配置位 CCP2MX 清零时，对 CCP2 进行其他分配。

表 **1-3：** **PIC18F8X90 I/O** 引脚说明 （续）

引脚名称	引脚号 TQFP	引脚类型	缓冲器类型	说明
				PORTB 是双向 I/O 口。PORTB 在所有输入端都可软件编程为内部弱上拉。
RB0/INT0	58			
RB0		I/O	TTL	数字 I/O。
INT0		I	ST	外部中断 0。
RB1/INT1/SEG8	57			
RB1		I/O	TTL	数字 I/O。
INT1		I	ST	外部中断 1。
SEG8		O	模拟	LCD 的 SEG8 输出。
RB2/INT2/SEG9	56			
RB2		I/O	TTL	数字 I/O。
INT2		I	ST	外部中断 2。
SEG9		O	模拟	LCD 的 SEG9 输出。
RB3/INT3/SEG10	55			
RB3		I/O	TTL	数字 I/O。
INT1		I	ST	外部中断 3。
SEG10		O	模拟	LCD 的 SEG10 输出。
RB4/KBI0/SEG11	54			
RB4		I/O	TTL	数字 I/O。
KBI0		I	TTL	电平变化中断引脚。
SEG11		O	模拟	LCD 的 SEG11 输出。
RB5/KBI1	53			
RB5		I/O	TTL	数字 I/O。
KBI1		I	TTL	电平变化中断引脚。
RB6/KBI2/PGC	52			
RB6		I/O	TTL	数字 I/O。
KBI2		I	TTL	电平变化中断引脚。
PGC		I/O	ST	在线调试器和 ICSP™ 编程时钟引脚。
RB7/KBI3/PGD	47			
RB7		I/O	TTL	数字 I/O。
KBI3		I	TTL	电平变化中断引脚。
PGD		I/O	ST	在线调试器和 ICSP 编程数据引脚。

图注： TTL = TTL 兼容输入　　CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器输入　　Analog = 模拟输入
I = 输入　　O = 输出
P = 电源　　OD = 漏极开路 （没有 P 型二极管接到 V_{DD}）

注 **1：** 当配置位 CCP2MX 置 1 时，对 CCP2 进行默认分配。
2： 当配置位 CCP2MX 清零时，对 CCP2 进行其他分配。

表 **1-3**： **PIC18F8X90 I/O** 引脚说明 （续）

引脚名称	引脚号 TQFP	引脚类型	缓冲器类型	说明
				PORTC 是双向 I/O 口。
RC0/T1OSO/T13CKI	36			
RC0		I/O	ST	数字 I/O。
T1OSO		O	—	Timer1 振荡器输出。
T13CKI		I	ST	Timer1/Timer3 外部时钟输入。
RC1/T1OSI/CCP2	35			
RC1		I/O	ST	数字 I/O。
T1OSI		I	CMOS	Timer1 振荡器输入。
CCP2[(1)]		I/O	ST	Capture2 输入 /Compare2 输出 /PWM2 输出
RC2/CCP1/SEG13	43			
RC2		I/O	ST	数字 I/O。
CCP1		I/O	ST	Capture1 输入 /Compare1 输出 /PWM1 输出。
SEG13		O	模拟	LCD 的 SEG13 输出。
RC3/SCK/SCL	44			
RC3		I/O	ST	数字 I/O。
SCK		I/O	ST	SPI™ 模式的同步串行时钟输入 / 输出。
SCL		I/O	ST	I^{2}C™ 模式的同步串行时钟输入 / 输出。
RC4/SDI/SDA	45			
RC4		I/O	ST	数字 I/O。
SDI		I	ST	SPI 数据输入。
SDA		I/O	ST	I^{2}C 数据 I/O。
RC5/SDO/SEG12	46			
RC5		I/O	ST	数字 I/O。
SDO		O	—	SPI 数据输出。
SEG12		O	模拟	LCD 的 SEG12 输出。
RC6/TX1/CK1	37			
RC6		I/O	ST	数字 I/O。
TX1		O	—	EUSART1 异步发送。
CK1		I/O	ST	EUSART1 同步时钟 （见 RX1/DT1 引脚信息）。
RC7/RX1/DT1	38			
RC7		I/O	ST	数字 I/O。
RX1		I	ST	EUSART1 异步接收。
DT1		I/O	ST	EUSART1 同步数据 （见 TX1/CK1 引脚信息）。

图注： TTL = TTL 兼容输入　　CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器输入　　Analog = 模拟输入
I = 输入　　O = 输出
P = 电源　　OD = 漏极开路 （没有 P 型二极管接到 VDD）

注 **1**： 当配置位 CCP2MX 置 1 时，对 CCP2 进行默认分配。
2： 当配置位 CCP2MX 清零时，对 CCP2 进行其他分配。

表 **1-3：** **PIC18F8X90 I/O** 引脚说明 （续）

引脚名称	引脚号 TQFP	引脚类型	缓冲器类型	说明
				PORTD 是双向 I/O 口。
RD0/SEG0	72			
RD0		I/O	ST	数字 I/O。
SEG0		O	模拟	LCD 的 SEG0 输出。
RD1/SEG1	69			
RD1		I/O	ST	数字 I/O。
SEG1		O	模拟	LCD 的 SEG1 输出。
RD2/SEG2	68			
RD2		I/O	ST	数字 I/O。
SEG2		O	模拟	LCD 的 SEG2 输出。
RD3/SEG3	67			
RD3		I/O	ST	数字 I/O。
SEG3		O	模拟	LCD 的 SEG3 输出。
RD4/SEG4	66			
RD4		I/O	ST	数字 I/O。
SEG4		O	模拟	LCD 的 SEG4 输出。
RD5/SEG5	65			
RD5		I/O	ST	数字 I/O。
SEG5		O	模拟	LCD 的 SEG5 输出。
RD6/SEG6	64			
RD6		I/O	ST	数字 I/O。
SEG6		O	模拟	LCD 的 SEG6 输出。
RD7/SEG7	63			
RD7		I/O	ST	数字 I/O。
SEG7		O	模拟	LCD 的 SEG7 输出。

图注： TTL = TTL 兼容输入　　CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器输入　　Analog = 模拟输入
I = 输入　　O = 输出
P = 电源　　OD = 漏极开路（没有 P 型二极管接到 VDD）

注 **1：** 当配置位 CCP2MX 置 1 时，对 CCP2 进行默认分配。
2： 当配置位 CCP2MX 清零时，对 CCP2 进行其他分配。

表 **1-3**： **PIC18F8X90 I/O** 引脚说明 （续）

引脚名称	引脚号 TQFP	引脚类型	缓冲器类型	说明
				PORTE 是双向 I/O 口。
LCDBIAS1	4			
LCDBIAS1		I	模拟	LCD 的 BIAS1 输入。
LCDBIAS2	3			
LCDBIAS2		I	模拟	LCD 的 BIAS2 输入。
LCDBIAS3	78			
LCDBIAS3		I	模拟	LCD 的 BIAS3 输入。
COM0	77			
COM0		O	模拟	LCD 的 COM0 输出。
RE4/COM1	76			
RE4		I/O	ST	数字 I/O。
COM1		O	模拟	LCD 的 COM1 输出。
RE5/COM2	75			
RE5		I/O	ST	数字 I/O。
COM2		O	模拟	LCD 的 COM2 输出。
RE6/COM3	74			
RE6		I/O	ST	数字 I/O。
COM3		O	模拟	LCD 的 COM3 输出。
RE7/CCP2/SEG31	73			
RE7		I/O	ST	数字 I/O。
CCP2(2)		I/O	ST	Capture 2 输入 /Compare 2 输出 /PWM 2 输出。
SEG31		O	模拟	LCD 的 SEG31 输出。

图注： TTL = TTL 兼容输入　　CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器输入　　Analog = 模拟输入
I = 输入　　O = 输出
P = 电源　　OD = 漏极开路 （没有 P 型二极管接到 VDD）

注 **1**： 当配置位 CCP2MX 置 1 时，对 CCP2 进行默认分配。
2： 当配置位 CCP2MX 清零时，对 CCP2 进行其他分配。

表 1-3： PIC18F8X90 I/O 引脚说明 （续）

引脚名称	引脚号 TQFP	引脚类型	缓冲器类型	说明
				PORTF 是双向 I/O 口。
RF0/AN5/SEG18	24			
RF0		I/O	ST	数字 I/O。
AN5		I	模拟	模拟输入 5。
SEG18		O	模拟	LCD 的 SEG18 输出。
RF1/AN6/C2OUT/SEG19	23			
RF1		I/O	ST	数字 I/O。
AN6		I	模拟	模拟输入 6。
C2OUT		O	—	Comparator 2 输出。
SEG19		O	模拟	LCD 的 SEG19 输出。
RF2/AN7/C1OUT/SEG20	18			
RF2		I/O	ST	数字 I/O。
AN7		I	模拟	模拟输入 7。
C1OUT		O	—	Comparator 1 输出。
SEG20		O	模拟	LCD 的 SEG20 输出。
RF3/AN8/SEG21	17			
RF3		I/O	ST	数字 I/O。
AN8		I	模拟	模拟输入 8。
SEG21		O	模拟	LCD 的 SEG21 输出。
RF4/AN9/SEG22	16			
RF4		I/O	ST	数字 I/O。
AN9		I	模拟	模拟输入 9。
SEG22		O	模拟	LCD 的 SEG22 输出。
RF5/AN10/CVREF/SEG23	15			
RF5		I/O	ST	数字 I/O。
AN10		I	模拟	模拟输入 10。
CVREF		O	模拟	比较器参考电压输出。
SEG23		O	模拟	LCD 的 SEG23 输出。
RF6/AN11/SEG24	14			
RF6		I/O	ST	数字 I/O。
AN11		I	模拟	模拟输入 11。
SEG24		O	模拟	LCD 的 SEG24 输出。
RF7/$\overline{SS}$/SEG25	13			
RF7		I/O	ST	数字 I/O。
$\overline{SS}$		I	TTL	SPI™ 从动选择输入。
SEG25		O	模拟	LCD 的 SEG25 输出。

图注： TTL = TTL 兼容输入　　CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器输入　　Analog = 模拟输入
I = 输入　　O = 输出
P = 电源　　OD = 漏极开路 （没有 P 型二极管接到 VDD）

注 **1：** 当配置位 CCP2MX 置 1 时，对 CCP2 进行默认分配。
2： 当配置位 CCP2MX 清零时，对 CCP2 进行其他分配。

表 **1-3**： **PIC18F8X90 I/O** 引脚说明 （续）

引脚名称	引脚号 TQFP	引脚类型	缓冲器类型	说明
				PORTG 是双向 I/O 口。
RG0/SEG30	5			
RG0		I/O	ST	数字 I/O。
SEG30		O	模拟	LCD 的 SEG30 输出。
RG1/TX2/CK2/SEG29	6			
RG1		I/O	ST	数字 I/O。
TX2		O	—	AUSART2 异步发送。
CK2		I/O	ST	AUSART2 同步时钟 （见相关的 RX2/DT2 引脚信息）。
SEG29		O	模拟	LCD 的 SEG29 输出。
RG2/RX2/DT2/SEG28	7			
RG2		I/O	ST	数字 I/O。
RX2		I	ST	AUSART2 异步接收。
DT2		I/O	ST	AUSART2 同步数据 （见相关的 TX2/CK2 引脚信息）。
SEG28		O	模拟	LCD 的 SEG28 输出。
RG3/SEG27	8			
RG3		I/O	ST	数字 I/O。
SEG27		O	模拟	LCD 的 SEG27 输出。
RG4/SEG26	10			
RG4		I/O	ST	数字 I/O。
SEG26		O	模拟	LCD 的 SEG26 输出。
RG5				请参见 $\overline{\text{MCLR}}$/VPP/RG5 引脚信息。

图注： TTL = TTL 兼容输入 CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器输入 Analog = 模拟输入
I = 输入 O = 输出
P = 电源 OD = 漏极开路 （没有 P 型二极管接到 VDD）

注 **1**： 当配置位 CCP2MX 置 1 时，对 CCP2 进行默认分配。
2： 当配置位 CCP2MX 清零时，对 CCP2 进行其他分配。

表 **1-3**： **PIC18F8X90 I/O** 引脚说明 （续）

引脚名称	引脚号 TQFP	引脚类型	缓冲器类型	说明
				PORTH 是双向 I/O 口。
RH0/SEG47	79			
RH0		I/O	ST	数字 I/O。
SEG47		O	模拟	LCD 的 SEG47 输出。
RH1/SEG46	80			
RH1		I/O	ST	数字 I/O。
SEG46		O	模拟	LCD 的 SEG46 输出。
RH2/SEG45	1			
RH2		I/O	ST	数字 I/O。
SEG45		O	模拟	LCD 的 SEG45 输出。
RH3/SEG44	2			
RH3		I/O	ST	数字 I/O。
SEG44		O	模拟	LCD 的 SEG44 输出。
RH4/SEG40	22			
RH4		I/O	ST	数字 I/O。
SEG40		O	模拟	LCD 的 SEG40 输出。
RH5/SEG41	21			
RH5		I/O	ST	数字 I/O。
SEG41		O	模拟	LCD 的 SEG41 输出。
RH6/SEG42	20			
RH6		I/O	ST	数字 I/O。
SEG42		O	模拟	LCD 的 SEG42 输出。
RH7/SEG43	19			
RH7		I/O	ST	数字 I/O。
SEG43		O	模拟	LCD 的 SEG43 输出。

图注： TTL = TTL 兼容输入　　CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器输入　　Analog = 模拟输入
I = 输入　　O = 输出
P = 电源　　OD = 漏极开路 （没有 P 型二极管接到 V_{DD}）

注 **1**： 当配置位 CCP2MX 置 1 时，对 CCP2 进行默认分配。
2： 当配置位 CCP2MX 清零时，对 CCP2 进行其他分配。

表 **1-3：** **PIC18F8X90 I/O** 引脚说明 （续）

引脚名称	引脚号 TQFP	引脚类型	缓冲器类型	说明
				PORTJ 是双向 I/O 口。
RJ0/SEG32	62			
RJ0		I/O	ST	数字 I/O。
SEG32		O	模拟	LCD 的 SEG32 输出。
RJ1/SEG33	61			
RJ1		I/O	ST	数字 I/O。
SEG33		O	模拟	LCD 的 SEG33 输出。
RJ2/SEG34	60			
RJ2		I/O	ST	数字 I/O。
SEG34		O	模拟	LCD 的 SEG34 输出。
RJ3/SEG35	59			
RJ3		I/O	ST	数字 I/O。
SEG35		O	模拟	LCD 的 SEG35 输出。
RJ4/SEG39	39			
RJ4		I/O	ST	数字 I/O。
SEG39		O	模拟	LCD 的 SEG39 输出。
RJ5/SEG38	40			
RJ5		I/O	ST	数字 I/O
SEG38		O	模拟	LCD 的 SEG38 输出。
RJ6/SEG37	41			
RJ6		I/O	ST	数字 I/O。
SEG37		O	模拟	LCD 的 SEG37 输出。
RJ7/SEG36	42			
RJ7		I/O	ST	数字 I/O。
SEG36		O	模拟	LCD 的 SEG36 输出。
VSS	11, 31, 51, 70	P	—	逻辑电路和 I/O 引脚的参考地。
VDD	12, 32, 48, 71	P	—	逻辑电路和 I/O 引脚的正向电源。
AVSS	26	P	—	模拟模块的参考地。
AVDD	25	P	—	模拟模块的正向电源。

图注：
- TTL = TTL 兼容输入
- CMOS = CMOS 兼容输入或输出
- ST = CMOS 电平的施密特触发器输入
- Analog = 模拟输入
- I = 输入
- O = 输出
- P = 电源
- OD = 漏极开路 （没有 P 型二极管接到 VDD）

注 **1：** 当配置位 CCP2MX 置 1 时，对 CCP2 进行默认分配。
2： 当配置位 CCP2MX 清零时，对 CCP2 进行其他分配。

注：

2.0 振荡器配置

2.1 振荡器类型

PIC18F6390/6490/8390/8490 器件可以在十种不同的振荡器模式下工作。通过编程配置寄存器 1H 中的配置位 FOSC3:FOSC0，用户可以选择这十种模式中的一种模式：

1. LP　低功耗晶振模式
2. XT　晶振 / 谐振器模式
3. HS　高速晶振 / 谐振器模式
4. HSPLL　使能 PLL 的高速晶振 / 谐振器振荡器模式
5. RC　外部电阻 / 电容振荡器模式，通过 RA6 引脚输出 Fosc/4 的信号
6. RCIO　外部电阻 / 电容振荡器模式，RA6 作为 I/O 引脚
7. INTIO1　内部振荡器模式，通过 RA6 引脚输出 Fosc/4 的信号， RA7 引脚为 I/O 引脚
8. INTIO2　内部振荡器模式， RA6 和 RA7 均用作 I/O 引脚
9. EC　带 Fosc/4 输出的外部时钟模式
10. ECIO　RA6 用作 I/O 引脚的外部时钟模式

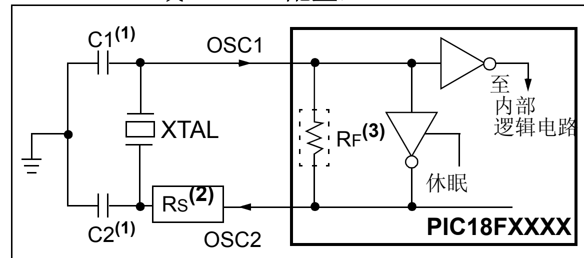
2.2 晶振 / 陶瓷谐振器

在 XT、LP、HS 或 HSPLL 振荡器模式下，晶振或陶瓷谐振器与 OSC1 和 OSC2 引脚相连来产生振荡。图 2-1 显示了引脚连接方式。

振荡器的设计要求使用平行切割的晶体。

注：	使用顺序切割的晶体，可能会使振荡器产生的频率超出晶体制造厂商所给的参数范围。

图 2-1：晶振 / 陶瓷谐振器工作原理 （XT、 LP、 HS 或 HSPLL 配置）



注 1： 如需了解 C1 和 C2 的初始值，请参见表 2-1 和表 2-2。
2： 对于AT条形切割的晶体可能会需要一个串联电阻 （Rs）。
3： RF 的值随选定的振荡器模式变化。

表 2-1：陶瓷谐振器的电容选择

使用的典型电容值：			
模式	频率	**OSC1**	**OSC2**
XT	455 kHz 2.0 MHz 4.0 MHz	56 pF 47 pF 33 pF	56 pF 47 pF 33 pF
HS	8.0 MHz 16.0 MHz	27 pF 22 pF	27 pF 22 pF

上述电容值仅供设计参考。

已在下列谐振器的基本起振和运行过程中测试了这些电容。**这些值并非最佳值。**

要得到理想的振荡器工作状况，可能需要不同的电容值。用户应当在设计的 VDD 和温度条件下测试振荡器的性能。

更多信息，请参见表 2-2 后面的 “注”。

使用的谐振器：	
455 kHz	4.0 MHz
2.0 MHz	8.0 MHz
16.0 MHz	

表 2-2： 晶振的电容选择

振荡类型	晶振频率	已测试的典型电容值：	
		C1	C2
LP	32 kHz	33 pF	33 pF
	200 kHz	15 pF	15 pF
XT	1 MHz	33 pF	33 pF
	4 MHz	27 pF	27 pF
HS	4 MHz	27 pF	27 pF
	8 MHz	22 pF	22 pF
	20 MHz	15 pF	15 pF

上述电容值仅供设计参考。

已在下列晶振的基本起振和运行过程中测试了这些电容。**这些值并非最佳值**。

要得到理想的振荡器工作状况，可能需要不同的电容值。用户应当在设计的 VDD 和温度条件下测试振荡器的性能

更多信息，请参见本表后面的 "注"。

所使用的晶振：	
32 kHz	4 MHz
200 kHz	8 MHz
1 MHz	20 MHz

注 1：电容值越大，振荡器的稳定性越高，但同时起振时间也越长。

2：当工作电压 VDD 低于 3V，或在任何电压下使用某些陶瓷谐振器时，可能需要使用 HS 振荡器模式或切换到晶振模式。

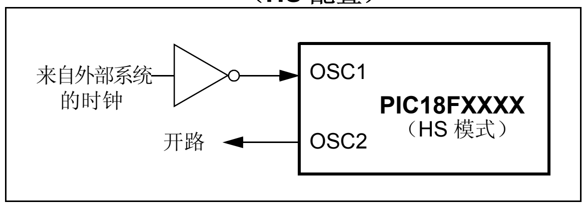
3：因为每种谐振器 / 晶振都有其自身特性，用户应当向谐振器 / 晶振制造厂商询问外部元件的适当值。

4：可能需要使用电阻 Rs 以避免对低驱动规格的晶体造成过驱动。

5：请始终验证在设计的 VDD 和温度范围下的振荡器性能。

如图 2-2 所示，在 HS 模式下，OSC1 引脚也可以连接外部时钟源。

图 2-2： 外部时钟输入工作原理（HS 配置）

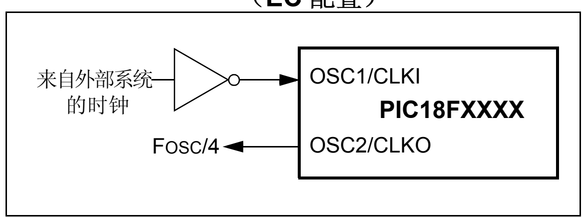


2.3 外部时钟输入

EC 和 ECIO 振荡器模式要求 OSC1 引脚与一个外部时钟源相连。在上电复位后或从休眠模式退出后，不需要振荡器起振时间。

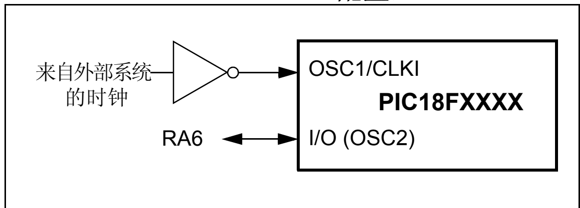
在 EC 振荡器模式下，由 OSC2 引脚输出振荡器频率的4分频信号。此信号可用于测试或同步其他逻辑。图 2-3 显示了 EC 振荡器模式的引脚连接方式。

图 2-3： 外部时钟输入工作原理（EC 配置）



ECIO 振荡器模式的工作方式类似于 EC 模式，不同之处在于 OSC2 引脚变成了一个额外的通用 I/O 引脚。该 I/O 引脚成为 PORTA 的第 6 位（RA6）。图 2-4 显示了 ECIO 振荡器模式下的引脚连接方式。

图 2-4： 外部时钟输入工作原理（ECIO 配置）



 初稿

2.4 RC 振荡器

对于对时序要求不高的应用，选择 RC 和 RCIO 器件能更好地节约成本。实际的振荡器频率是以下几个因素的函数：

- 供电电压
- 外部电阻 （REXT）和电容 （CEXT）的值
- 工作温度

给定同样的器件、工作电压和温度以及元件值，振荡的频率仍然会各不相同。这些频率上的差异是由诸如以下因素引起的：

- 正常制造工艺的差异
- 不同封装类型引线电容的不同 （尤其当 CEXT 值较小时）
- REXT 和 CEXT 在容限范围内的数值波动

在 RC 振荡器模式下，由 OSC2 引脚输出振荡器频率的 4分频信号。此信号可用于测试或同步其他逻辑。图 2-5 显示了外接 R/C 组合电路的连接方式。

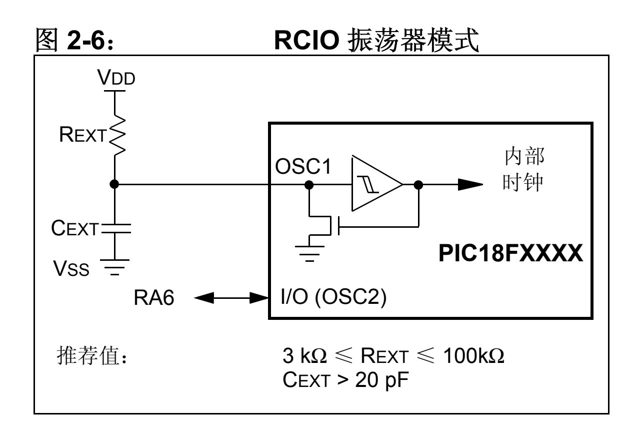
图 2-5： RC 振荡器模式

VDD
REXT
CEXT
VSS
OSC1
内部时钟
PIC18FXXXX
OSC2/CLKO
FOSC/4

推荐值： 3kΩ ≤ REXT ≤ 100 kΩ
CEXT > 20 pF

RCIO 振荡器模式 （图 2-6）的工作方式类似于 RC 模式，不同之处在于 OSC2 引脚变成了一个额外的通用 I/O 引脚。该 I/O 引脚成为 PORTA 的第 6 位 （RA6）。

图 2-6： RCIO 振荡器模式



2.5 PLL 倍频器

如果用户希望使用低频晶振电路或通过晶振将器件频率调节至其最高额定频率，可以选择使用锁相环 （PLL）电路。对于担心高频晶振引起 EMI 或需要内部振荡器提供高速时钟的用户而言，这样做可能有用。

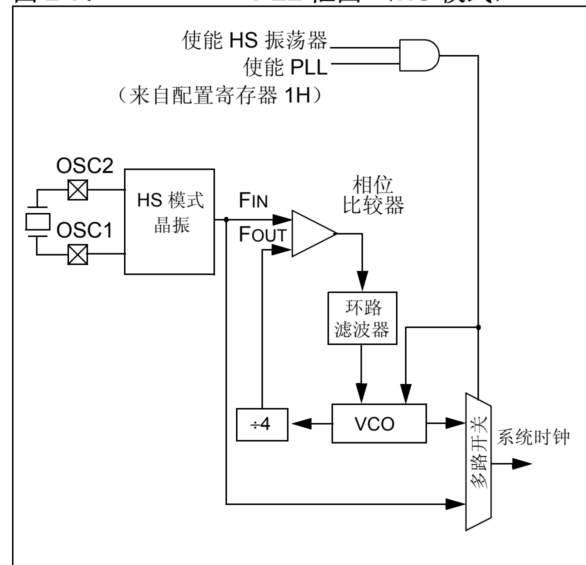
2.5.1 HSPLL 振荡器模式

HSPLL 模式使用 HS 模式振荡器产生最高 10 MHz 的频率。然后 PLL 将振荡器输出频率乘以 4，从而产生最高 40 MHz 的内部时钟频率。

仅当将 FOSC3:FOSC0 配置位编程为 HSPLL 模式（=0110）时，晶振才可以使用 PLL。



图 2-7： PLL 框图 （HS 模式）



2.5.2 PLL 和 INTOSC

在选定的振荡器模式下，内部振荡器电路也可以使用 PLL。在此配置下， PLL 用软件使能并产生最高为 32 MHz 的时钟输出。第 **2.6.4** 节 **“INTOSC 模式下的 PLL”**描述了使用 PLL 的 INTOSC 的工作原理。

2.6 内部振荡器电路

PIC18F6390/6490/8390/8490 器件含有可产生两种不同时钟信号的内部振荡器电路。这两种信号均可充当单片机的时钟源，从而避免在 OSC1 和 / 或 OSC2 引脚上使用外部振荡器电路。

主输出（INTOSC）是一个 8 MHz 的时钟源，可以用于直接驱动器件时钟。它还可以驱动后分频器，该分频器可提供从 31 kHz 到 4 MHz 的时钟频率。当选择了 125 kHz 到 8 MHz 的时钟频率时，使能 INTOSC 输出。

另一个时钟源是内部 RC 振荡器（INTRC），它提供了标称值为 31 kHz 的输出。如果选择 INTRC 作为器件的时钟源，它就会被使能；当使能以下任一功能时，也将自动使能 INTRC：

- 上电延迟定时器
- 故障保护时钟监视器
- 看门狗定时器
- 双速启动
- 将 INTRC 作为时钟源的 LCD

第 **23.0 节 "CPU 的特殊功能"**详细讨论以上功能。

通过配置 OSCCON 寄存器（寄存器 2-2）的 IRCF 位，可以选择时钟源频率（INTOSC 直接频率、 INTRC 直接频率或 INTOSC 后分频器频率）。

2.6.1 INTIO 模式

使用内部振荡器作为时钟源可以避免使用两个外部振荡器引脚，而将它们用作数字 I/O。目前有两种不同的配置：

- 在 INTIO1 模式下， OSC2 引脚输出 Fosc/4，而 OSC1 引脚充当 RA7，用于数字输入和输出。
- 在 INTIO2 模式下，OSC1 充当 RA7，OSC2 充当 RA6，两者都用于数字输入和输出。

2.6.2 INTOSC 输出频率

出厂时已校准了内部振荡器电路使之能够产生 8.0 MHz 的 INTOSC 输出频率。

INTRC 振荡器的工作独立于 INTOSC 源。电压和温度变化导致的 INTOSC 变化并不一定会使 INTRC 变化，反之亦然。

2.6.3 OSCTUNE 寄存器

内部振荡器的输出已在出厂前经过校准，但仍可以在用户应用中被调整。这是通过写 OSCTUNE 寄存器（寄存器 2-1）完成的。在整个调节范围内调节灵敏度保持不变。

当修改了 OSCTUNE 寄存器后，INTOSC 和 INTRC 的频率将变化为新的频率。INTRC 时钟将在 8 个时钟周期（大约 8 * 32μs = 256 μs）内达到新的频率。 INTOSC 时钟会在 1 ms 内稳定下来。在此变化期间，代码会继续执行。不会有任何迹象表明时钟频率发生了改变。

OSCTUNE 寄存器也有 INTSRC 和 PLLEN 位，它们控制内部振荡器电路的某些功能。当选择了 31 kHz 频率后，用户可通过 INTSRC 位选择用作时钟源的内部振荡器。在第 **2.7.1 节"振荡器控制寄存器"**中对此进行了更详细地说明。

在内部振荡器模式下， PLLEN 位控制 PLL 倍频器的工作模式。

2.6.4 INTOSC 模式下的 PLL

内部振荡器电路可以通过使用 4x 倍频器来产生比一般内部振荡器所能产生的时钟速度更快的器件时钟速度。当使能时， PLL 最高可产生 32 MHz 的时钟速度。

与 HSPLL 模式不同，PLL 由软件控制。控制位 PLLEN（OSCTUNE<6>）用来使能或禁止其工作。

当器件被配置为使用内部振荡器电路作为其主时钟源时（FOSC3:FOSC0 = `1001` 或 `1000`），可以使用 PLL。此外，仅当选定的输出频率是 4 MHz 或 8 MHz（OSCCON<6:4> = `111` or `110`）时，PLL 才会工作。如果两个条件都不满足，则会禁止 PLL。

PLLEN 控制位只有在使用 PLL 的内部振荡器模式下才有效。在所有其他模式下，它被强制为 `0` 并且无效。

2.6.5 INTOSC 频率漂移

厂家校准内部振荡器电路的输出（INTOSC）为 8 MHz。但是，此频率可能会随着 VDD 电压或温度的改变而发生漂移，这一点可能会以各种方式影响控制器的运行。通过修改 OSCTUNE 寄存器的值可以调节 INTOSC 的频率。这不会对 INTRC 时钟源的频率造成影响。

调节 INTOSC 时钟源需要了解何时调节、调节的方向以及在某些情况下的调整量。第 **2.6.5.1 节"用 AUSART 进行补偿"**、第 **2.6.5.2 节"用定时器进行补偿（1）"**和第 **2.6.5.3 节"用定时器进行补偿（2）"**讨论了三种补偿技术，但是也可能使用其他的技术。

2.6.5.1 用 AUSART 进行补偿

当 AUSART 开始产生帧错误，或者在异步模式下接收数据有错误时可能需要进行调节。帧错误表示器件时钟的频率太高。要对此进行调节，可以减小 OSTUNE 寄存器中的值来降低时钟频率。另一方面，数据中有错误可能表明时钟速度太低。要进行补偿，可以增大 OSTUNE 寄存器中的值来提高时钟频率。

2.6.5.2 用定时器进行补偿 （1）

此技术是将器件时钟的速度与某些参考时钟进行比较。可能要用到两个定时器：一个由外设时钟提供时钟源，而另一个由一个固定的参考源 （如 Timer1 振荡器）提供时钟源。

两个定时器都被清零，但由参考源提供时钟信号的定时器产生中断。当中断发生时，使用内部时钟源的定时器值被读取且两个定时器均被清零。如果使用内部时钟源的定时器的值大于期望值，则表示内部振荡器电路运行过快。要对此进行调整，需减小 OSCTUNE 寄存器中的值。

2.6.5.3 用定时器进行补偿 （2）

CCP 模块可以使用由内部振荡器电路提供时钟信号的独立运行的 Timer1 （或 Timer3）和已知周期的外部事件 （即 AC 电源频率）。在 CCPRxH:CCPRxL 寄存器中捕获并记录第一个事件的时间。当第二个事件导致捕捉时，要用第二个事件的时间减去第一个事件的时间。由于外部事件的周期是已知的，因此可以计算事件之间的时间差。

如果测得的时间比计算得到的时间大很多，则表示内部振荡器电路运行过快。要对此进行补偿，需减小 OSTUNE寄存器中的值。如果测得的时间比计算得到的时间小得多，则表示内部振荡器电路运行过慢。需增大 OSTUNE 寄存器中的值来补偿。

寄存器 2-1： OSCTUNE：振荡器调节寄存器

R/W-0	R/W-0(1)	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
INTSRC	PLLEN(1)	—	TUN4	TUN3	TUN2	TUN1	TUN0
bit 7							bit 0

bit 7 **INTSRC**：内部振荡器低频源选择位
1 = 来自 8 MHz INTOSC 源的31.25 kHz 器件时钟 （使能 256 分频）
0 = 直接来自 INTRC 内部振荡器的31 kHz 器件时钟

bit 6 **PLLEN**：INTOSC 的倍频器 PLL 使能位 (1)
1 = 为 INTOSC 使能PLL （仅 4 MHz 和 8 MHz）
0 = 禁止 PLL

注 1：仅在某些振荡器配置中可用；其他情况下，此位不可用，并且读为 0。详细信息请参见第 **2.6.4 节 "INTOSC 模式下的 PLL"**。

bit 5 **未用位**：读为 0

bit 4-0 **TUN4:TUN0**：频率调节位
01111 = 最高频率
• •
• •
00001
00000 = 中心频率。振荡器模块运行在已校准的频率上。
11111
• •
• •
10000 = 最低频率

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

2.7 时钟源与振荡器切换

与以前的 PIC18 器件一样，PIC18F6390/6490/8390/8490 系列包含允许将器件时钟源从主振荡器切换到备用低频时钟源的功能。PIC18F6390/6490/8390/8490 器件提供了两个备用时钟源。当使能备用时钟源时，可以使用多种功耗管理工作模式。

基本上，这些器件都有 3 种时钟源：

- 主振荡器
- 辅助振荡器
- 内部振荡器电路

主振荡器包括外部晶振和谐振器模式、外部 RC 模式、外部时钟模式和内部振荡器电路。特定的模式由 FOSC3:FOSC0 控制位定义。这些模式的详细信息已在本章前面的内容中作过介绍。

辅助振荡器是不与 OSC1 或 OSC2 引脚连接的外部时钟源。即使在控制器处于功耗管理模式时这些时钟源仍可继续工作。

PIC18F6390/6490/8390/8490 器件将 Timer1 振荡器作为辅助振荡器。此振荡器 （在所有的功耗管理模式中）通常是实时时钟等功能的时基。

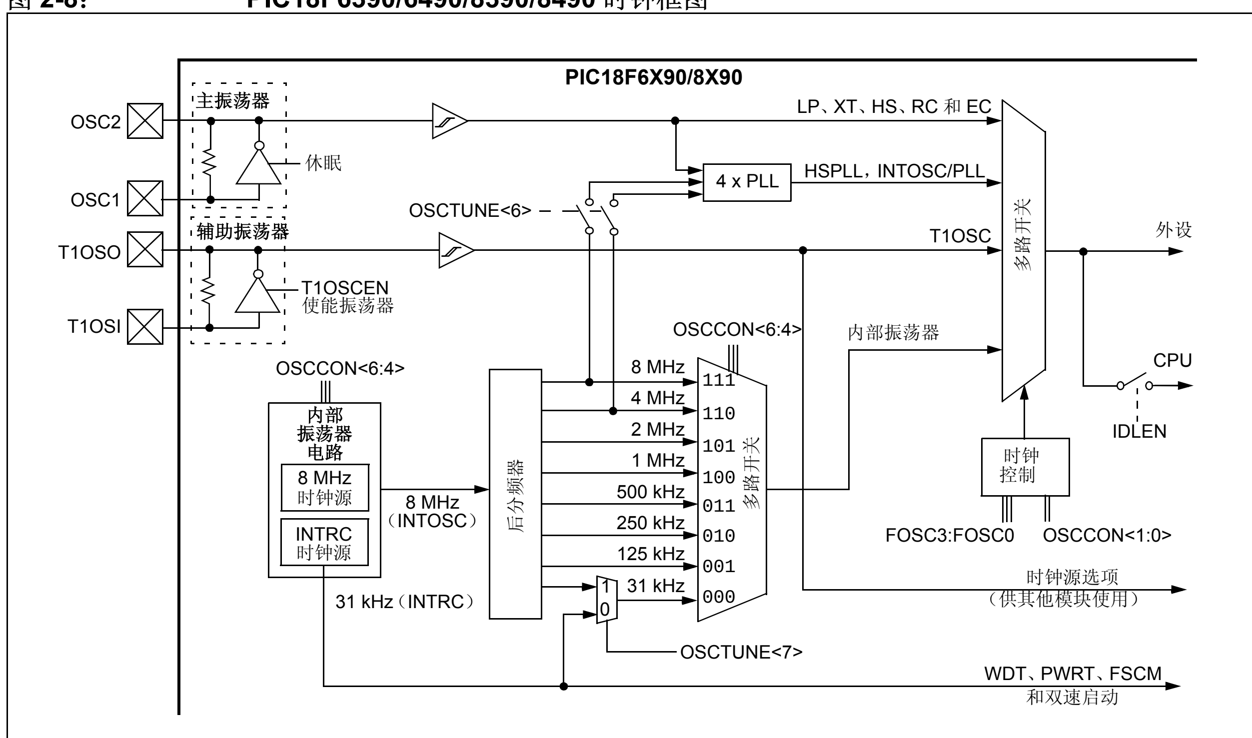
大部分情况下，在 RC0/T1OSO/T13CKI 和 RC1/T1OSI 引脚之间接有一个 32.768 kHz 的时钟晶振。与 LP 模式振荡器电路类似，在每个引脚与地之间均接有负载电容。

将在**第 11.3 节 “Timer1 振荡器”**中详细讨论 Timer1 振荡器。

除了作为主时钟源之外，**内部振荡器电路**还可以作为功耗管理模式的时钟源。INTRC 源也可作为几种特殊功能部件 （例如 WDT 和故障保护时钟监视器）的时钟源。

在图 2-8 显示了 PIC18F6390/6490/8390/8490 器件的时钟源。如需了解配置寄存器的详细信息，请参见**第 23.0 节 “CPU 的特殊功能”**。

图 2-8： PIC18F6390/6490/8390/8490 时钟框图



2.7.1 振荡器控制寄存器

OSCCON 寄存器（寄存器 2-2）控制全功耗模式和功耗管理模式下器件时钟工作的多个方面。

系统时钟选择位 SCS1:SCS0 用于选择时钟源。可用的时钟源包括主时钟（由 FOSC:FOSC0 配置位定义）、辅助时钟（Timer1 振荡器）和内部振荡器电路。当写入一个或多个位之后，接着是一段很短的时钟转换间隔，然后时钟源会立即改变。在所有形式的复位中 SCS 位都会被清零。

内部振荡器频率选择位 IRCF2:IRCF0 选择内部振荡器电路的输出频率。这些频率可以是 INTRC 源的频率、INTOSC 源的频率（8 MHz）或 INTOSC 后分频器产生的几个频率之一（31.25 kHz 到 4 MHz）。如果器件时钟由内部振荡器电路提供，改变这些位的状态会使内部振荡器输出立即改变。

当选定了 31 kHz 的输出频率（IRCF2:IRCF0 = 000）时，用户可以选择用作时钟源的内部振荡器。这通过使用OSCTUNE寄存器中的INTSRC位（OSCTUNE<7>）完成。将该位置 1 选择 INTOSC 作为时钟源，并通过使能 INTOSC 后分频器的 256 分频输出，使该时钟源输出 31.25 kHz 的时钟信号。将 INTSRC 位清零选择 INTRC（标称值为 31 kHz）作为时钟源。

此选项使用户能选择可调节且更精确的INTOSC作为时钟源，同时以非常低的时钟速度运行以节省功耗。无论 INTSRC 的设置如何，INTRC 总是作为看门狗定时器和故障保护时钟监视器之类部件的时钟源。

OSTS、IOFS 和 T1RUN 位指出当前提供器件时钟的是哪一个时钟源。OSTS 位置 1 表明振荡器起振定时器已超时且主时钟在主时钟模式下作为器件时钟。IOFS 位置 1 表明内部振荡器电路已稳定并在 RC 时钟模式下提供器件时钟。T1RUN 位（T1CON<6>）置 1 表明 Timer1 振荡器正在辅助时钟模式下提供器件时钟。在功耗管理模式下，任何时候这 3 个位中只有一个会置 1。如果这些位都没有置 1，则表示当前时钟源是 INTRC，或内部振荡器电路刚刚起振且尚未稳定。

IDLEN 位决定当执行 SLEEP 指令时器件是进入休眠模式还是某个空闲模式。

第 3.0 节 “功耗管理模式” 更详细地讨论了 OSCCON 寄存器中标志和控制位的使用。

注		
	1：	要选择辅助时钟源，必须使能 Timer1 振荡器。通过将 Timer1 控制寄存器中的 T1OSCEN 位（T1CON<3>）置 1，可以使能 Timer1 振荡器。如果未使能 Timer1 振荡器，则在执行 SLEEP 指令期间选择辅助时钟源的任何尝试都会被忽略。
	2：	建议在Timer1振荡器稳定工作之后再执行SLEEP指令，否则当Timer1振荡器起振时可能会发生很长的延迟。

2.7.2 振荡器转换

PIC18F6390/6490/8390/8490 器件包含了防止在切换时钟源时发生时钟“毛刺”的电路。在切换时钟时，系统时钟会有短暂的停顿。该停顿的时间长度是旧时钟源的两个周期加上新时钟源的三到四个周期的和。此公式假设新时钟源是稳定的。

第 3.1.2 节“进入功耗管理模式”详细讨论了时钟转换。

寄存器 **2-2**： **OSCCON**：振荡器控制寄存器

R/W-0	R/W-1	R/W-0	R/W-0	R[(1)]	R-0	R/W-0	R/W-0
IDLEN	IRCF2	IRCF1	IRCF0	OSTS	IOFS	SCS1	SCS0

bit 7 bit 0

bit 7 **IDLEN**：空闲使能位

1 = 执行 SLEEP 指令后器件进入空闲模式
0 = 执行 SLEEP 指令后器件进入休眠模式

bit 6-4 **IRCF2:IRCF0**：内部振荡器频率选择位

111 = 8 MHz （由 INTOSC 直接驱动时钟）
110 = 4 MHz
101 = 2 MHz
100 = 1 MHz [(3)]
011 = 500 kHz
010 = 250 kHz
001 = 125 kHz
000 = 31 kHz （来自 INTOSC/256 或直接来自 INTRC[(2)]）

bit 3 **OSTS**：振荡器起振延时状态位 [(1)]

1 = 振荡器起振定时器延时已结束；主振荡器正在运行
0 = 振荡器起振定时器延时正在进行；主振荡器尚未就绪

bit 2 **IOFS**：INTOSC 频率稳定位

1 = INTOSC 频率已稳定
0 = INTOSC 频率尚未稳定

bit 1-0 **SCS1:SCS0**：系统时钟选择位

1x = 内部振荡器电路
01 = Timer1 振荡器
00 = 主振荡器

注 **1**：取决于 IESO 配置位的状态。

2：时钟源由 INTSRC 位 （OSCTUNE<7>）选择，请参见第 **2.6.3** 节 **“OSCTUNE 寄存器”**。

3：复位时 INTOSC 的默认输出频率。

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

2.8 功耗管理模式对各种时钟源的影响

当选定了 PRI_IDLE 模式后，指定的主振荡器会继续运行而不中断。对于所有其他的功耗管理模式，使用 OSC1 引脚的振荡器会被禁止。OSC1 引脚（以及由振荡器使用的 OSC2 引脚）将会停止振荡。

在辅助时钟模式下（SEC_RUN和SEC_IDLE），Timer1 振荡器作为器件时钟源工作。如果需要，Timer1 振荡器也可以运行在所有功耗管理模式下为 Timer1 或 Timer3 提供时钟。

在内部振荡器模式下（RC_RUN 和 RC_IDLE），由内部振荡器电路提供器件时钟。无论是哪种功耗管理模式，31 kHz 的 INTRC 输出均可被直接用来提供时钟并且可被使能来支持多种特殊的功能部件（如需了解更多有关 WDT、故障保护时钟监视器和双速启动的信息，请参见**第 23.2 节“看门狗定时器（WDT）”**到**第 23.4 节“故障保护时钟监视器”**）。8 MHz 的 INTOSC 输出可以直接用于为器件提供时钟，或者也可先由后分频器进行分频再用作器件时钟。如果直接由 INTRC 输出提供时钟，则会禁止 INTOSC 输出。

如果选择了休眠模式，所有的时钟源都会被停止。因为休眠模式切断了所有晶体管的开启电流，休眠模式能实现最小的器件电流消耗（仅泄漏电流）。

在休眠期间使能任何片上功能都将增加休眠时的电流消耗。要支持 WDT 工作，需要使能 INTRC。Timer1 振荡器可以用来为实时时钟提供时钟源。不需要器件时钟源的其他功能部件也可以工作（即，SSP 从动器件和 INTn 引脚等）。在**第 26.2 节“DC 特性：掉电和供电电流”**中列出了可能显著增加电流消耗的外设。

2.9 上电延迟

由两个定时器控制上电延迟，这样大多数应用都无需外接复位电路。上电延迟可以确保在器件电源稳定（常规环境下）和主时钟稳定工作之前器件保持复位状态。如需更多有关上电延迟的信息，请参见**第 4.5 节“器件复位定时器”**。

第一个定时器是上电延迟定时器（PWRT），在上电时它提供了固定的延迟时间（表 26-10 中的参数 33）。通过清零 PWRTEN 配置位可使能它。

第二个定时器是振荡器起振定时器（OST），用于在晶振稳定前使芯片保持在复位状态（LP、XT 和 HS 模式）。OST 通过计数 1024 个振荡周期后允许振荡器为器件提供时钟。

当选定 HSPLL 振荡器模式时，器件将在 HS 模式下的 OST延迟之后另外保持2 ms的复位状态，这样可使PLL 锁定输入时钟频率。

POR 之后有一个 T_{CSD} 间隔的延迟（表 26-10 中的参数 38），在延迟期间控制器为执行指令做准备。此延迟紧随其他延迟之后。任将 EC、RC 或 INTIO 模式之一用作主时钟源时，这可能是唯一的延迟。

表 2-3： 休眠模式下 OSC1 和 OSC2 引脚的状态

振荡器模式	OSC1 引脚	OSC2 引脚
RC， INTIO1	悬空，经外部电阻上拉为高电平	处于逻辑低电平 （时钟 /4 输出）
RCIO， INTIO2	悬空，经外部电阻上拉为高电平	配置为 PORTA 的第 6 位
ECIO	悬空，由外部时钟拉高	配置为 PORTA 的第 6 位
EC	悬空，由外部时钟拉高	处于逻辑低电平 （时钟 /4 输出）
LP、 XT 和 HS	反馈反相器被禁用，处于静止电平	反馈反相器被禁用，处于静止电平

注： 关于由休眠和 $\overline{\text{MCLR}}$ 复位引起的超时，请参见**第 4.0 节“复位”**中的表 4-2。

注：

3.0 功耗管理模式

PIC18F6390/6490/8390/8490 器件总共提供七种工作模式，可以更有效的进行功耗管理。这些工作模式提供了多种选择，可在资源受限的应用（即，电池供电的器件）中节省功耗。

功耗管理模式有三种类别：

- 休眠模式
- 空闲模式
- 运行模式

这些类别定义了需要为器件的哪些部分提供时钟源，有时还需要定义时钟源的速度。运行和空闲模式可以使用三种时钟源（主时钟源、辅助时钟源或 INTOSC 复用器）中的任意一种；而休眠模式则不能使用时钟源。

功耗管理模式包括几个节省功耗的功能。其中之一就是其他 PIC18 器件也提供的时钟切换功能，该功能允许使用 Timer1 振荡器代替主振荡器。节省功耗的功能还包括所有 PICmicro® 器件都提供的休眠模式，所有器件的时钟都在休眠模式停止。

3.1 选择功耗管理模式

选择功耗管理模式需要决定是否为 CPU 提供时钟源以及选择何种时钟源。 IDLEN 位控制是否为 CPU 提供时钟源，而 SC1:SCS0 位选择时钟源。表 3-1 总结了各个模式下的位设置、时钟源和受影响的模块。

3.1.1 时钟源

SCS1:SCS0 位允许在功耗管理模式的三个时钟源中进行选择。它们是：

- 主时钟，由 FOSC3:FOSC0 配置位定义
- 辅助时钟 （Timer1 振荡器）
- 内部振荡电路 （用于 RC 模式）

3.1.2 进入功耗管理模式

可以通过装载 OSCCON 寄存器可进入功耗管理运行模式，或从一种功耗管理模式切换到另一种功耗管理模式。SCS1:SCS0 位选择时钟源并确定使用运行模式还是空闲模式。更改这些位会导致立即切换到一个新的时钟源 （假定新时钟源正在运行）。此切换可能会引起时钟转换延迟。**第 3.1.3 节 “时钟转换和状态指示”**及其后续章节将会讨论这些问题。

执行SLEEP指令可以触发进入功耗管理空闲模式或休眠模式。最后实际进入哪个模式由 IDLEN 状态位决定。

更改功耗管理模式并不总是要求设置所有的位，而是取决于当前的模式和将要切换到的模式。通过在执行SLEEP指令之前更改振荡器选择位或更改IDLEN位可完成多种模式转换。如果已经正确配置了 IDLEN 位，可能只须执行 SLEEP 指令就可实现模式切换。

表 **3-1**： 功耗管理模式

模式	OSCCON 位		模块时钟		可用时钟和振荡器
	IDLEN(1) <7>	SCS1:SCS0 <1:0>	CPU	外设	
休眠	0	N/A	关闭	关闭	无——所有时钟被禁止
PRI_RUN	N/A	00	为其提供时钟信号	为其提供时钟信号	主时钟——LP， XT， HS， HSPLL， RC， EC， INTRC(2)：这是正常的全功耗执行模式。
SEC_RUN	N/A	01	为其提供时钟信号	为其提供时钟信号	辅助时钟——Timer1 振荡器
RC_RUN	N/A	1x	为其提供时钟信号	为其提供时钟信号	内部振荡器模块 (2)
PRI_IDLE	1	00	关闭	为其提供时钟信号	主时钟——LP、 XT、 HS、 HSPLL、 RC 和 EC
SEC_IDLE	1	01	关闭	为其提供时钟信号	辅助时钟——Timer1 振荡器
RC_IDLE	1	1x	关闭	为其提供时钟信号	内部振荡器模块 (2)

注 **1**： IDLEN 反映它在执行 SLEEP 指令时的值。
2： 包含 INTOSC 和 INTOSC 后分频器，以及 INTRC 源。

3.1.3 时钟转换和状态指示

在两个时钟源之间进行转换所需的时间长度是旧时钟源的两个周期与新时钟源的三到四个周期的和。此公式假设新时钟源是稳定的。

以下三位用于指明当前的时钟源及其状态。它们是：

- OSTS (OSCCON<3>)
- IOFS (OSCCON<2>)
- T1RUN (T1CON<6>)

一般来说，在一个给定的功耗管理模式中，这三个位中只有一个位会置 1。当 OSTS 位置 1 时，表明由主时钟提供器件时钟。当 IOFS 位置 1 时，表明由 INTOSC 输出提供 8 MHz 的时钟源到分频器，实际上由分频器驱动器件时钟。当 T1RUN 位置 1 时，表明由 Timer1 振荡器提供时钟源。如果这些位均不置 1，则表明要么由 INTRC 时钟源为器件提供时钟信号，要么 INTOSC 源尚未稳定。

如果用FOSC3:FOSC0配置位将内部振荡电路配置为主时钟源，则在 PRI_RUN 或 PRI_IDLE 模式中， OSTS 和 IOFS 位可能同时置 1。这表示主时钟 （INTOSC 输出）正在产生稳定的 8 MHz 输出。进入工作频率相同的另一个 RC 功耗管理模式将清零 OSTS 位。

注	
1：	在仅修改 IRCF 位时应该特别小心。如果 VDD 电压小于 3V，可以选择比低 VDD 电压所能支持的时钟速度更高的速度。违反 VDD/FOSC 规范会导致器件运行不正常。
2：	执行SLEEP指令并不一定会将器件置于休眠模式。它只是作为触发条件，让器件进入休眠模式或一种空闲模式，具体何种模式由 IDLEN 位的设置决定。

3.1.4 多种休眠命令

使用SLEEP指令调用功耗管理模式时，其模式在该指令执行那一刻由 IDLEN 位的设置决定。如果执行了另一条 SLEEP 指令，器件将在这时进入由 IDLEN 位指定的功耗管理模式。如果 IDLEN 位已更改，器件将进入由新的设置指定的新的功耗管理模式。

3.2 运行模式

在运行模式中，内核和外设的时钟都是激活的。这些运行模式之间的区别就在于时钟源的不同。

3.2.1 PRI_RUN 模式

PRI_RUN 模式是单片机的正常全功耗执行模式。除非使能了双速启动，该模式也是器件复位后的默认模式（详情请见**第 23.3 节 "双速启动"**）。在此模式中，OSTS 位置 1。如果内部振荡电路为主时钟源，IOFS 位也可能置 1 （见**第 2.7.1 节 "振荡器控制寄存器"**）。

3.2.2 SEC_RUN 模式

SEC_RUN 模式与其他 PIC18 器件提供的"时钟切换"功能兼容。在此模式下， CPU 和外设将 Timer1 振荡器作为时钟源。这允许用户在使用高精度时钟源的情况下仍可获得较低的功耗。

通过将 SCS1:SCS0 位置为 01 进入 SEC_RUN 模式。器件时钟源被切换到 Timer1 振荡器（见图 3-1），主振荡器被关闭， T1RUN 位 （T1CON<6>）被置 1 并且 PSTS 位被清零。

注：	Timer1 振荡器应该在进入 SEC_RUN 模式之前就已经运行了。如果在SCS1:SCS0位被置为 01 时 T1OSCEN 位没有置 1，就不会进入 SEC_RUN 模式。如果 Timer1 振荡器已经被使能，但没有开始运行，外设时钟将会延迟直到该振荡器起振。在这种情况下，最初的振荡器运行很不稳定，可能会导致无法预料的结果。

在从 SEC_RUN 模式转换到 PRI_RUN 模式时，外设和 CPU 继续使用 Timer1 振荡器作为时钟源，直到主时钟启动。当主时钟准备好以后，时钟切换回主时钟 （见图 3-2）。当时钟切换完成后，T1RUN位被清零，OSTS 位被置 1 并且由主时钟提供器件时钟。这种唤醒不会影响 IDLEN 和 SCS 位。 Timer1 振荡器继续运行。

图 3-1： 进入 SEC_RUN 模式的转换时序

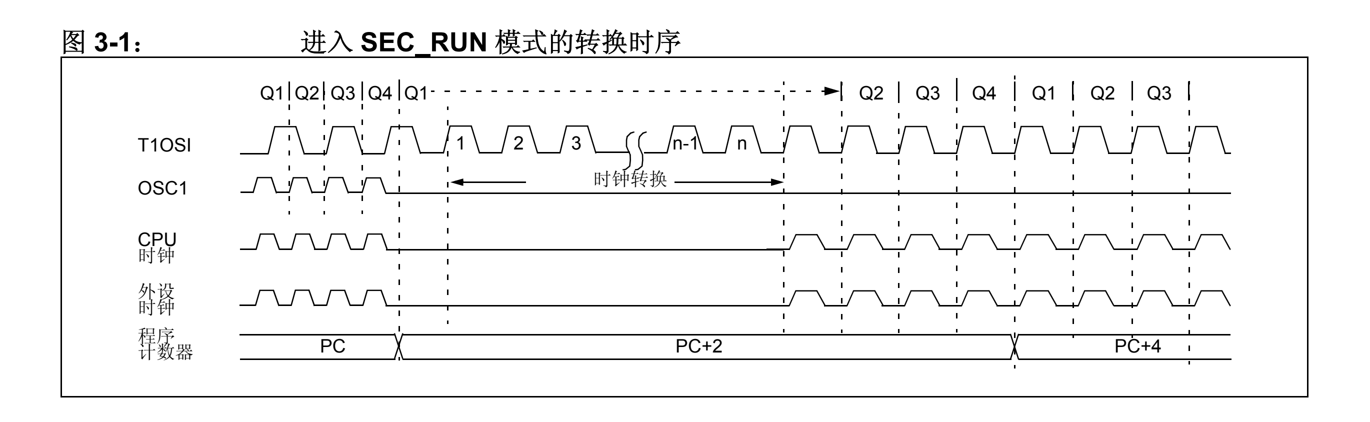
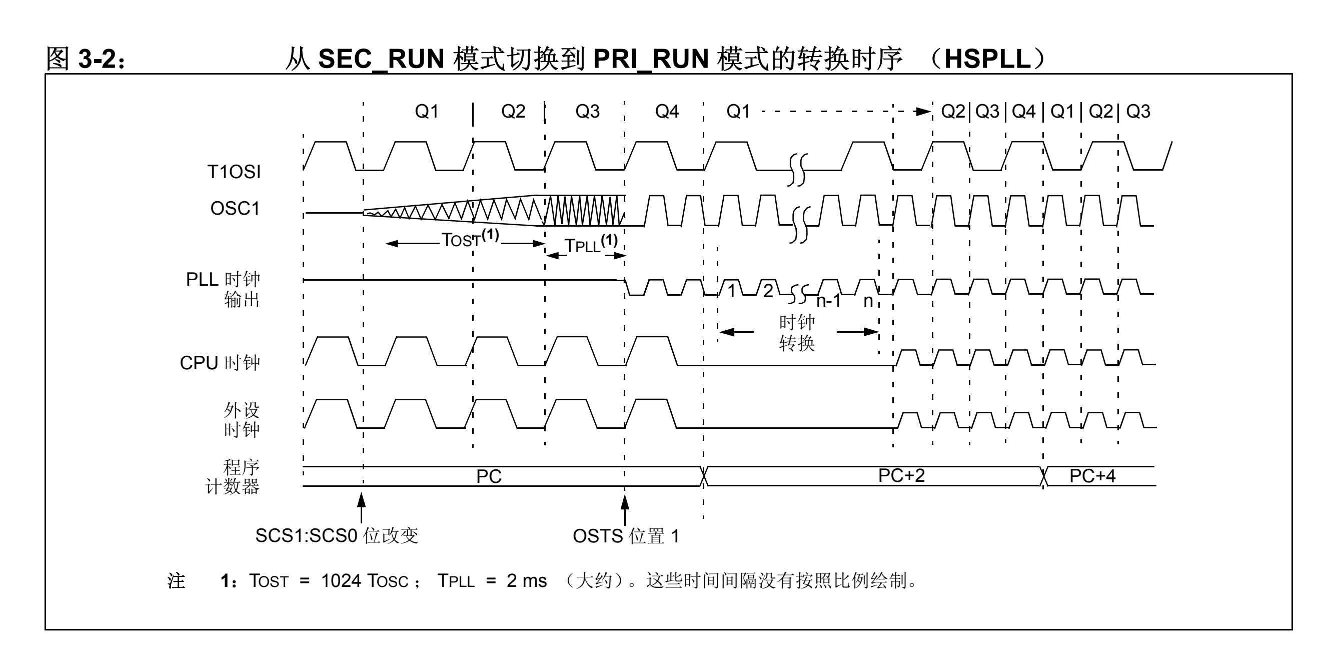


图 3-2： 从 SEC_RUN 模式切换到 PRI_RUN 模式的转换时序 （HSPLL）



注 1：T_{OST} = 1024 T_{OSC}； T_{PLL} = 2 ms （大约）。这些时间间隔没有按照比例绘制。

3.2.3 RC_RUN 模式

RC_RUN 模式中，内部振荡电路使用 INTOSC 复用器作为 CPU 和外设的时钟源，此时主时钟关闭。在使用 INTRC 时钟源时，此模式是在代码执行期间所有运行模式中最节省功耗的运行模式，它非常适用于对定时精度要求不高或者不是一直需要高速时钟的应用。

如果主时钟源为内部振荡电路（INTIO1 或 INTIO2），在代码执行期间， PRI_RUN 和 RC_RUN 这两种模式区别不大。但是在进入和退出 RC_RUN 模式时会发生时钟切换时间延迟。因此，如果主时钟源为内部振荡电路，建议不要使用 RC_RUN 模式。

通过将 SCS1 位置 1 可以进入此模式。虽然常常被忽略，但还是建议将 SCS0 位清零，从而保证软件的兼容性。当将时钟源切换到 INTOSC 复用器 （见图 3-3），主振荡器将被关闭并且 OSTS 位被清零。在任何时候更改 IRCF 位可以立即更改时钟速度。

注：	在仅修改 IRCF 位时应该特别小心。如果 VDD 电压小于 3V，可以选择比低 VDD 电压所能支持的时钟速度更高的速度。违反 VDD/FOSC 规范会导致器件运行不正常。

如果 IRCF 位和 INTSRC 位均被清零将禁止 INTOSC 输出并且 IOFS 位将保持清零。此时不会有关于当前时钟源的任何指示。由 INTRC 时钟源提供器件时钟。

如果 IRCF 位从全清零状态改变（因而使能 INTOSC 输出），或者 INTSRC 被置 1，在 INTOSC 输出稳定后 IOFS 位将被置 1。在一个 TIOBST 间隔之后， INTOSC 时钟源趋于稳定，此时器件时钟继续运行。

如果之前的 IRCF 为一个非零值，或者在设置 SCS1 之前 INTSRC 已经置 1 并且 INTOSC 已达到稳定，那么 IOFS 位将保持置 1 状态。

在从 RC_RUN 模式转换到 PRI_RUN 模式时，在主时钟处于启动状态时，器件将继续使用 INTOSC 复用器作为时钟源。当主时钟准备好以后，时钟切换回主时钟（见图 3-4）。当时钟切换完成后， IOFS 位被清零，OSTS 位被置 1 并且由主时钟提供器件时钟。这种切换不会影响 IDLEN 和 SCS 位。如果使能了 WDT 或故障保护时钟监视器， INTRC 源将继续运行。

图 3-3： 到 RC 模式的转换时序

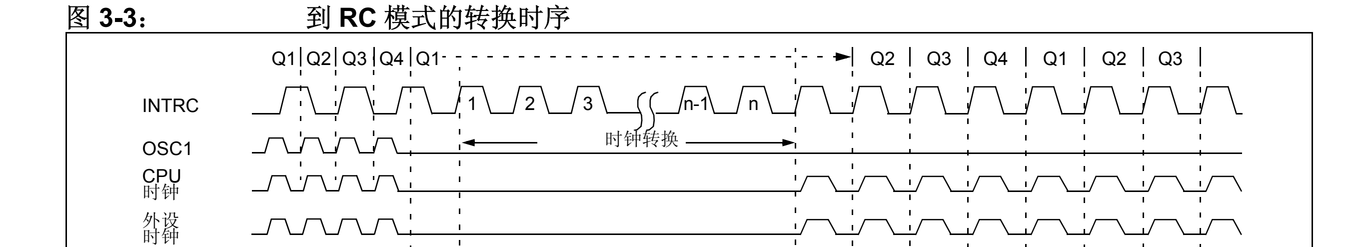
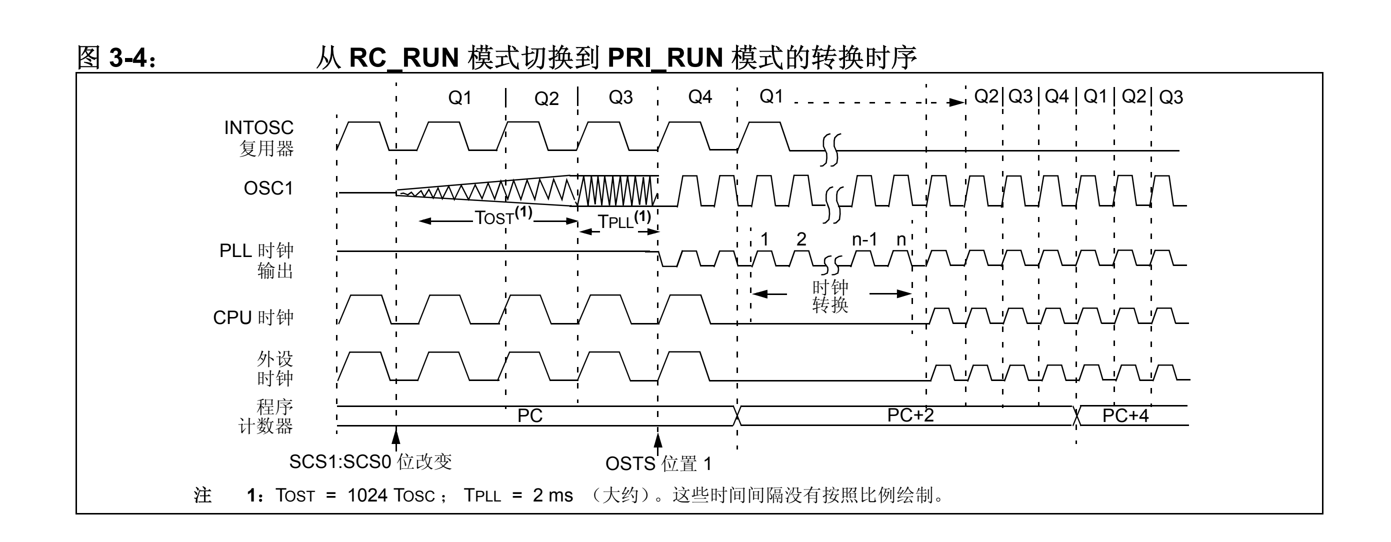


图 3-4： 从 RC_RUN 模式切换到 PRI_RUN 模式的转换时序



注 1：TOST = 1024 TOSC； TPLL = 2 ms （大约）。这些时间间隔没有按照比例绘制。

3.3 休眠模式

PIC18F6390/6490/8390/8490 器件的功耗管理休眠模式和所有其他PICmicro器件提供的传统休眠模式相同。通过清零 IDLEN 位（器件复位时的默认状态）并执行 SLEEP指令即可进入此模式。这将关闭所选择的振荡器（见图 3-5）并将所有的时钟源状态位清零。

从其他模式进入休眠模式不需要时钟切换。这是因为单片机一旦进入休眠模式就不需要时钟了。如果选择了 WDT，INTRC 时钟源将继续工作。如果使能了 Timer1 振荡器， Timer1 也将继续运行。

当在休眠模式中发生唤醒事件时（通过中断、复位或 WDT 超时），在主时钟准备好之前器件将没有时钟源（见图 3-6），或者如果使能了双速启动或故障保护时钟监视器，它将使用内部振荡电路作为时钟源（见第 **23.0** 节 **"CPU 的特殊功能"**）。在这两种情况下，当由主时钟提供器件时钟时， OSTS 位将置 1。这种唤醒不会影响 IDLEN 和 SCS 位。

3.4 空闲模式

空闲模式允许在外设继续工作的时候有选择的关闭单片机的 CPU。选择特定的空闲模式允许用户进一步管理功耗。

如果在执行 SLEEP 指令时，IDLEN 位被置为"1"，外设将使用由 SCS1:SCS0 位选择的时钟作为时钟源，而 CPU 没有时钟源。时钟源状态位不受影响。将 IDLEN 置 1 并执行 SLEEP 指令可以从给定的运行模式快速切换到相应的空闲模式。

如果选择了 WDT， INTRC 时钟源将继续工作。如果使能了 Timer1 振荡器， Timer1 也将继续运行。

由于 CPU 没有执行指令，器件只能通过中断、WDT 超时或复位从空闲模式退出。当发生唤醒事件时，CPU 会在其准备好执行代码前延时一个 TCSD 间隔（表 26-10 中的参数 38）。当 CPU 开始执行代码时，它将沿用与当前空闲模式相同的时钟源。例如，当从 RC_IDLE 模式唤醒时，将使用内部振荡电路为 CPU 和外设提供时钟（即 RC_RUN 模式）。这种唤醒不会影响 IDLEN 和 SCS 位。

当处于任何空闲模式或休眠模式中时， WDT 超时会导致 WDT 唤醒并进入当前由 SCS1:SCS0 位指定的运行模式。

图 3-5： 进入休眠模式的时序转换

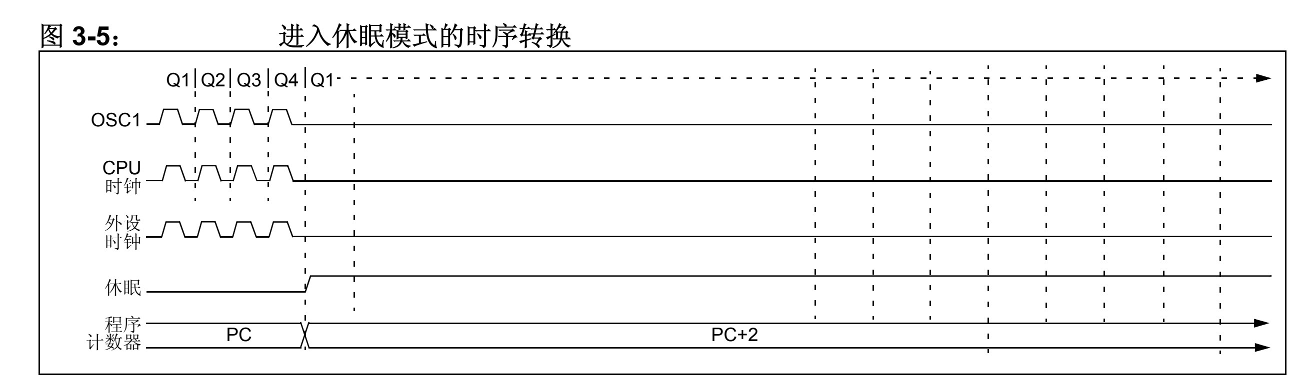
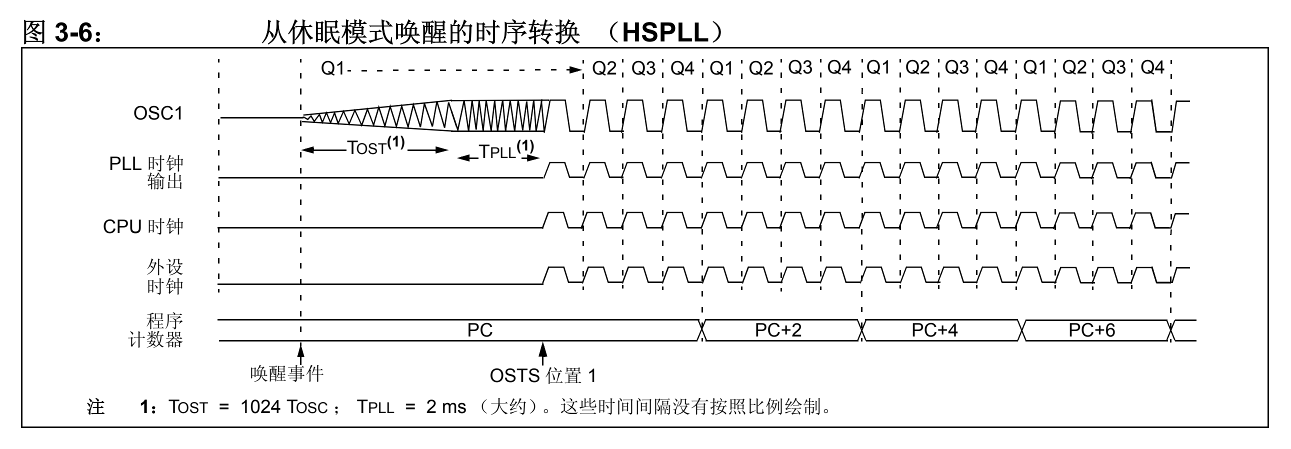


图 3-6： 从休眠模式唤醒的时序转换 （**HSPLL**）



注 1：TOST = 1024 TOSC； TPLL = 2 ms（大约）。这些时间间隔没有按照比例绘制。

3.4.1 PRI_IDLE 模式

在三种低功耗空闲模式中，只有该模式不会禁止主器件时钟。由于时钟源不需要“热身”或是从其他振荡器转换，选用此模式可以使对时间要求较高的应用以最快的速度恢复器件运行并使用较精确的主时钟源。

可以通过将 IDLEN 位置 1 并执行 SLEEP 指令以实现从 PRI_RUN 模式进入 PRI_IDLE 模式。如果器件在另一种运行模式，可以先将 IDLEN 位置 1，然后清零 SCS 位并执行 SLEEP。虽然 CPU 已被禁止，但外设仍可继续使用由FOSC3:FOSC0配置位指定的主时钟源为其提供时钟信号。OSTS 位保持置位 （见图 3-7）。

当发生唤醒事件时，由主时钟源为CPU 提供时钟。在唤醒事件和代码执行开始之间需要一个 TCSD 间隔的延迟。该延迟用来让 CPU 做好执行指令的准备。在唤醒之后，OSTS 位保持置 1 状态。这种唤醒不会影响 IDLEN 和 SCS 位 （见图 3-8）。

图 3-7： 进入 PRI_IDLE 模式的转换时序

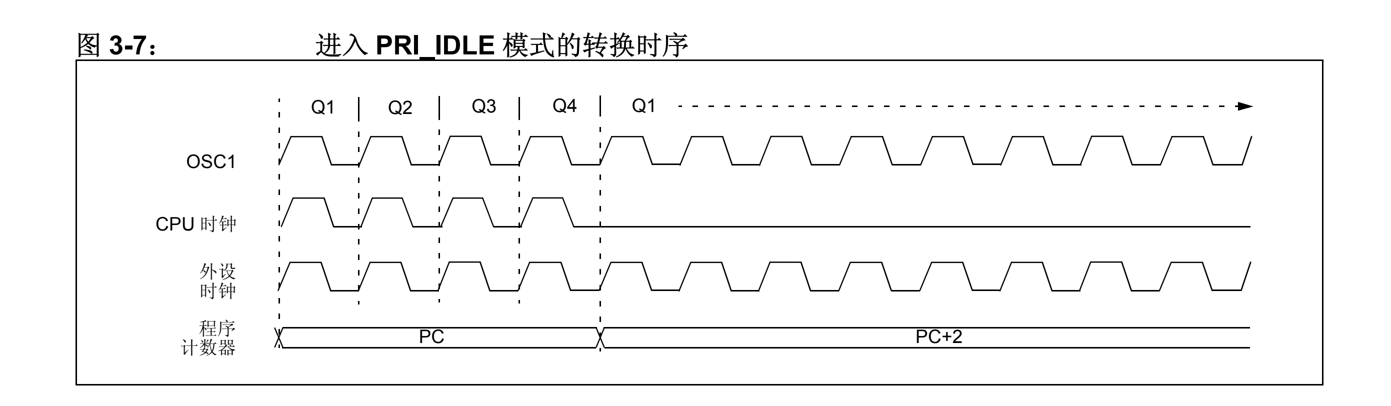
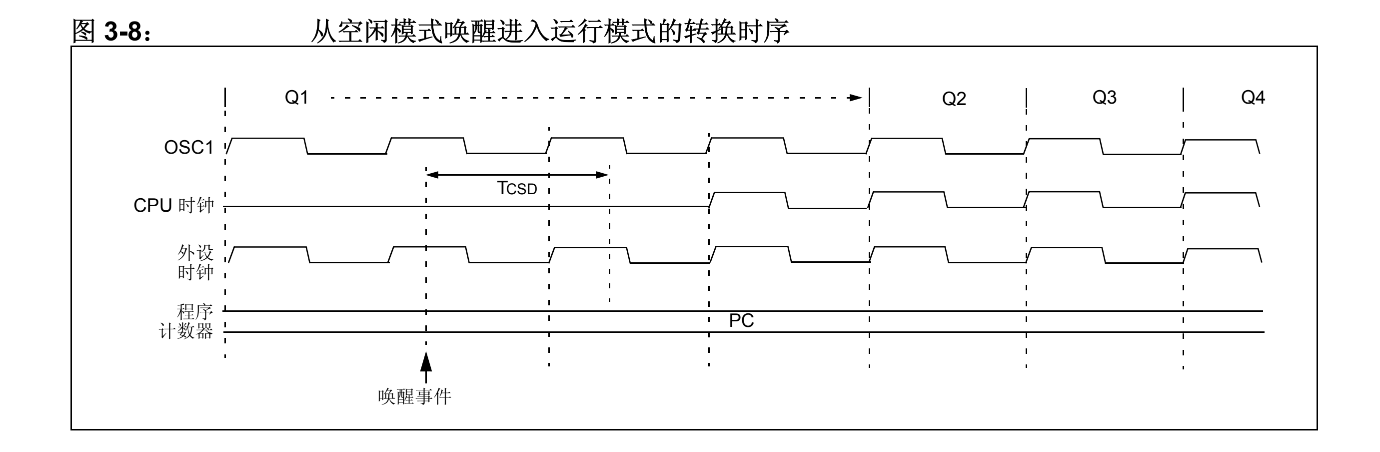


图 3-8： 从空闲模式唤醒进入运行模式的转换时序



 初稿

3.4.2 SEC_IDLE 模式

在 SEC_IDLE 模式中， CPU 被禁止，但外设继续将 Timer1 振荡器作为时钟源。可以通过将 IDLEN 位置 1 并执行 SLEEP 指令从 SEC_RUN 模式进入此模式。如果器件处于另一种运行模式，首先将 IDLEN 位置 1，然后将 SCS1:SCS0 置为 “01” 并执行 SLEEP。当时钟源切换到 Timer1 振荡器时，主振荡器关闭， OSTS 位被清零并且 T1RUN 位置 1。

当唤醒事件发生时，外设继续将 Timer 振荡器作为时钟源。唤醒事件发生后经过一个 T_{CSD} 时间间隔，CPU 开始执行代码并使用 Timer1 振荡器作为其时钟源。这种唤醒不会影响 IDLEN 和 SCS 位。Timer1 振荡器继续运行 （见图 3-8）。

注： Timer1 振荡器应该在进入 SEC_RUN 模式之前就已经运行了。如果执行 SLEEP 指令时 T1OSCEN 位没有置 1，就会忽略 SLEEP 指令并不会进入 SEC_IDLE 模式。如果使能了 Timer1 振荡器，但它尚未运行，外设时钟将会延迟直到该振荡器起振。在这种情况下，最初的振荡器运行很不稳定，可能会导致无法预料的结果。

3.4.3 RC_IDLE 模式

RC_IDLE 模式禁止 CPU，但仍继续由使用 INTOSC 复用器的内部振荡电路为外设提供时钟。该模式允许在空闲期间对功耗进行控制。

可以通过将IDLEN位置1并执行SLEEP指令从RC_RUN 模式进入此模式。如果器件处于另一种运行模式，可以先将 IDLEN 位置 1，然后再将 SCS1 位置 1 并执行 SLEEP。虽然 SCS0 的值常常被忽略，但仍建议将其清零，这将保证软件的兼容性。通过在执行 SLEEP 指令之前修改 IRCF 位可以使用 INTOSC 复用器来选择更高的时钟频率。当时钟源切换到 INTOSC 复用器时，主振荡器被关闭， OSTS 位被清零。

如果 IRCF 位被设置为任何非零值，或者 INTSRC 位被置 1，就会使能 INTOSC 输出。在一个 T_{IOBST} 间隔（表 26-10 中的参数 39）之后 INTOSC 将输出趋于稳定，随后 IOFS 位置 1。外设的时钟继续运行直到 INTOSC 时钟源趋于稳定。如果之前的 IRCF 为一个非零的值或者在执行 SLEEP 指令之前 INTSRC 已置 1，并且当前 INTOSC 源已经稳定，IOFS 位将保持置 1 状态。如果IRCF和INTSRC位全部清零，就不会使能INTOSC 输出， IOFS 位将保持清零状态，此时将不会有当前时钟源的任何指示。

当唤醒事件发生时，外设继续将 INTOSC 复用器作为时钟源。在唤醒事件后的 T_{CSD} 间隔之后，CPU 开始执行代码并使用 INTOSC 复用器作为时钟源。这种唤醒不会影响 IDLEN 和 SCS 位。如果使能了 WDT 或故障保护时钟监视器， INTRC 源将继续运行。

3.5 退出空闲和休眠模式

由中断、复位或 WDT 超时触发从休眠模式或任何空闲模式的退出。本节将讨论从功耗管理模式退出的触发方式。在每种功耗管理模式中我们还将讨论其时钟源子系统的作用（见第 **3.2** 节 **“运行模式”**到第 **3.4** 节 **“空闲模式”**）。

3.5.1 通过中断退出

任何可用的中断源都可导致器件从空闲模式或休眠模式退出到运行模式。要使能此功能，必须通过在某个 INTCON 或 PIE 寄存器中将中断源的使能位置 1 来使能中断源。当相应的中断标志位置 1 时，触发退出操作。

当使用中断从空闲或休眠模式退出时，如果 GIE/GIEH 位（INTCON<7>）置 1，程序就会跳转到中断矢量处执行代码。否则代码就会顺序执行（见第 **8.0** 节 **“中断”**）。

唤醒事件之后需要一个固定的 T_{CSD} 间隔的延迟，器件才会退出休眠和空闲模式。CPU 需要此延迟来准备执行代码。在延迟后的第一个时钟周期重新开始执行指令。

3.5.2 通过 WDT 超时退出

WDT 超时根据超时发生时器件所处的不同功耗管理模式会进行不同的操作。

如果器件不在执行代码（所有空闲模式和休眠模式），超时将导致从功耗管理模式退出（见第 **3.2** 节**“运行模式”**和第 **3.3** 节 **“休眠模式”**）。如果器件正在执行代码（所有运行模式），超时将导致 WDT 复位（见第 **23.2** 节 **“看门狗定时器（WDT）”**）。

执行 `SLEEP` 或 `CLRWDT` 指令、当前选择的时钟源失效（如果使能了故障保护监视器）以及如果器件时钟源为内部振荡电路，修改 OSCCON 寄存器中的 IRCF 位，均将清零 WDT 定时器和后分频器。

3.5.3 通过复位退出

通常，器件通过振荡器起振定时器（OST）保持在复位状态，直到主时钟就绪。主时钟就绪后，OSTS 位置 1，器件开始执行代码。如果以内部振荡电路作为新的时钟源，则 IOFS 位将置 1。

从复位状态退出到开始执行代码期间的延迟时间由唤醒前后的时钟源以及如果新的时钟源为主时钟，还有主时钟振荡器的类型决定。表 3-2 为退出延迟汇总。

可以在主时钟就绪之前开始执行代码。如果使能了双速启动（见第 **23.3** 节 **“双速启动”**）或故障保护时钟监视器（见第 **23.4** 节**“故障保护时钟监视器”**），器件可以在清零复位源的时候就开始执行代码。由内部振荡电路驱动的 INTOSC 复用器作为代码执行的时钟源。执行代码时，由内部振荡电路提供时钟源直到主时钟就绪，或者在主时钟就绪前进入功耗管理模式，随后将关闭主时钟。

3.5.4 在没有振荡器起振延迟的情况下退出

从某些功耗管理模式退出完全不需要 OST 延迟。有以下两种情形：

- 主时钟源不停止的 PRI_IDLE 模式和
- 主时钟源不是 LP、 XT、 HS 或 HSPLL 中的任意一种模式。

在这些情况下，主时钟源不需要振荡器起振延迟，因为它已经在运行（PRI_IDLE），或者它本来就不需要振荡器起振延迟（RC、EC 和 INTIO 振荡器模式）。但是，当器件退出休眠和空闲模式时，在唤醒事件之后仍然需要一个固定的 T_{CSD} 间隔的延迟，以便让 CPU 准备好执行代码。在延迟后的第一个时钟周期重新开始执行指令。

表 **3-2**： 通过复位从休眠模式或空闲模式唤醒的退出延迟 （按时钟源分类）

唤醒之前的时钟源	唤醒之后的时钟源	退出延迟	时钟准备状态位（**OSCCON**）
主器件时钟（PRI_IDLE 模式）	LP, XT, HS	TCSD[2]	OSTS
	HSPLL		
	EC, RC, INTRC[1]		—
	INTOSC[3]		IOFS
T1OSC 或 INTRC[1]	LP, XT, HS	TOST[4]	OSTS
	HSPLL	TOST + t_{rc}[4]	
	EC, RC, INTRC[1]	TCSD[2]	—
	INTOSC[2]	TIOBST[5]	IOFS
INTOSC[3]	LP, XT, HS	TOST[5]	OSTS
	HSPLL	TOST + t_{rc}[4]	
	EC, RC, INTRC[1]	TCSD[2]	—
	INTOSC[2]	无	IOFS
无（休眠模式）	LP, XT, HS	TOST[4]	OSTS
	HSPLL	TOST + t_{rc}[4]	
	EC, RC, INTRC[1]	TCSD[2]	—
	INTOSC[2]	TIOBST[5]	IOFS

注 **1**： 这种情况是特别针对 31 kHz INTRC 时钟源的。

2： 当从休眠模式和空闲模式唤醒时都需要 TCSD （参数 38）延迟，该延迟与任何所需的其他延迟并存 （见第 **3.4** 节 **“空闲模式”**）。

3： 包括 INTOSC 8 MHz 时钟源和后分频器产生的频率。

4： TOST 是振荡器起振定时器的延迟时间 （参数 32）。 t_{rc} 是 PLL 锁定延时定时器的延迟时间 （参数 F12）。后者也被称为 TPLL。

5： 在 TIOBST （参数 39）延迟和 INTOSC 稳定过程中，代码继续执行。

 初稿

注：

4.0 复位

PIC18F6390/6490/8390/8490 器件有以下几种不同的复位方式：

a） 上电复位 （Power-on Reset， POR）
b） 正常工作状态下的 MCLR 复位
c） 功耗管理模式下的 MCLR 复位
d） 看门狗定时器 （WDT）复位 （执行程序期间）
e） 可编程欠压复位 （Brown-out Reset， BOR）
f） RESET 指令
g） 堆栈满复位
h） 堆栈下溢复位

本节将讨论由 MCLR、POR 和 BOR 产生的复位，并涉及各种起振定时器的运行。堆栈复位事件将在第 **5.1.2.4 节“堆栈满和下溢复位”**中讨论。WDT 复位将在第 **23.2 节 “看门狗定时器 （WDT）”** 中讨论。

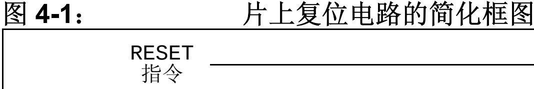
图 4-1 显示了片上复位电路的简化框图。

4.1 RCON 寄存器

通过 RCON 寄存器（寄存器 4-1）跟踪器件复位事件。该寄存器的低 5 位表明是否已经发生了特定的复位事件。在大多数情况下，只能通过事件将这些位置 1，而且必须由该事件后的应用程序将它们清零。这些标志位的状态，放在一起，可以被读取以表明刚发生的复位的类型。在第 **4.6 节“寄存器的复位状态”**中对此进行了更详细地说明。

RCON 寄存器中还有用于设置中断优先级的控制位（IPEN）和用于对 BOR 进行软件控制的控制位（SBOREN）。在第 **8.0 节 “中断”**中讨论了中断优先级。在第 **4.4 节“欠压复位（BOR）”**中讨论了 BOR。

图 4-1： 片上复位电路的简化框图



注 1： 这是来自内部振荡器模块的 INTRC 时钟源，它与 CLKI 引脚上的 RC 振荡器无关。
2： 超时情况见表 4-2。

寄存器 4-1： RCON：复位控制寄存器

R/W-0	R/W-1(1)	U-0	R/W-1	R-1	R-1	R/W-0	R/W-0
IPEN	SBOREN	—	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$
bit 7							bit 0

bit 7 **IPEN**：中断优先级使能位
1 = 使能中断优先级
0 = 禁止中断优先级 （PIC16CXXX 兼容模式）

bit 6 **SBOREN**：BOR 软件使能位
如果 BOREN1:BOREN0 = 01：
1 = 使能 BOR
0 = 禁止 BOR
如果 BOREN1:BOREN0 = 00、 10 或 11：
该位被禁止并读为 0。

注 1：如果使能了 SBOREN 位，其复位状态为 “1” ；否则为 “0”。

bit5 未用位：读为 0

bit4 **$\overline{RI}$**：RESET 指令标志位
1 = 未执行 RESET 指令 （只能由硬件置 1）
0 = 已执行了 RESET 指令，导致器件复位 （必须在发生欠压复位后由软件置 1）

bit3 **$\overline{TO}$**：看门狗超时标志位
1 = 通过上电、 CLRWDT 指令或 SLEEP 指令置 1。
0 = 发生了 WDT 超时溢出

bit2 **$\overline{PD}$**：掉电检测标志位
1 = 通过上电或 CLRWDT 指令置 1
0 = 通过执行 SLEEP 指令清零

bit1 **$\overline{POR}$**：上电复位状态位
1 = 未发生上电复位 （只能由硬件置 1）
0 = 已发生上电复位 （必须在发生上电复位后由软件置 1）

bit0 **$\overline{BOR}$**：欠压复位状态位
1 = 未发生欠压复位 （只能由硬件置 1）
0 = 已发生欠压复位 （必须在发生欠压复位后由软件置 1）

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

注 1：建议在检测到上电复位后，将 $\overline{POR}$ 位置 1，以便继续检测后续的上电复位。
2：当 $\overline{BOR}$ 为 0 并且 POR 为 1 时（假定在 POR 之后立即用软件将 $\overline{POR}$ 置 1），可以说已发生了欠压复位。

4.2 主清零 （$\overline{\text{MCLR}}$）

$\overline{\text{MCLR}}$ 引脚提供了用外部硬件触发器件复位的方法。将该引脚拉低可以产生复位信号。PIC18 扩展型 MCU 器件在 $\overline{\text{MCLR}}$ 复位路径上有一个噪声滤波器，该滤波器检测并滤除小的干扰脉冲。

任何内部复位，包括 WDT 复位，均不能将 $\overline{\text{MCLR}}$ 引脚驱动为低电平。

在 PIC18F6390/6490/8390/8490 器件中，可以用 MCLRE 配置位禁止 $\overline{\text{MCLR}}$ 输入。当禁止 $\overline{\text{MCLR}}$ 时，该引脚将成为一个数字输入引脚。如需更多信息，请参见**第 9.7 节 “PORTG、 TRISG 和 LATG 寄存器”**。

4.3 上电复位 （POR）

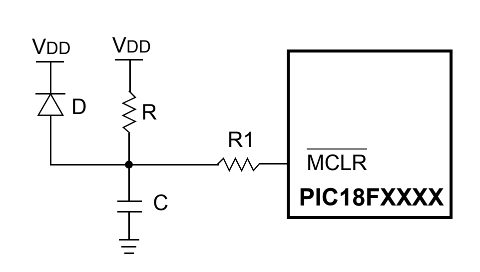
只要当 VDD 上升到某个门限时，就会在片上产生上电复位脉冲。这使得 VDD 达到满足器件正常工作的数值时，器件会以初始化状态起动。

为了利用 POR 电路，将 $\overline{\text{MCLR}}$ 引脚通过一个电阻（阻值范围为 1 kΩ 到 10 kΩ）连接到 VDD。这样可以省去产生上电复位延时通常所需的外部 RC 元件。VDD 的最小上升速率已指定 （参数 D004）。上升速率缓慢的情况，请参见图 4-2。

当器件开始正常工作 （即，退出复位状态）时，器件的工作参数（电压、频率和温度等）必须得到满足，以确保其正常工作。如果不满足这些条件，那么器件必须保持在复位状态，直到满足工作条件为止。

POR 事件由 $\overline{\text{POR}}$ 位 （RCON<1>）捕获。每当发生 POR 时，该位的状态就会被置为 0；任何其他的复位事件均不能改变它。任何硬件事件均不能将 POR 复位为 1。要捕获多个事件，用户必须在 POR 之后用软件手动地将该位复位为 1。

图 4-2： 外部上电复位电路 （VDD 缓慢上电的情况）



注 1： 仅当 VDD 上电速率过慢时才需要外部上电复位电路。二极管 D 有助于在 VDD 掉电时使电容迅速放电。

2： 建议 R < 40 kΩ，确保电阻 R 两端压降符合器件的电气规范。

3： R1 ≥ 1 kΩ 将限制任何电流从外部电容 C 流入 $\overline{\text{MCLR}}$，以避免由于静电放电（Electrostatic Discharge，ESD）或电超载（Electrical Overstress，EOS）导致 $\overline{\text{MCLR}}$/VPP 引脚损坏。

4.4 欠压复位 （BOR）

PIC18F6390/6490/8390/8490 器件带有一个 BOR 电路，它将为用户提供一系列配置和节能选项。BOR 由 BORV1:BORV0 和 BOREN1:BOREN0 配置位控制。总共有四种BOR配置，归纳在表 4-1中。

BOR 门限值由 BORV1:BORV0 位设置。如果使能了 BOR（BOREN1:BOREN0 为除“00”以外的任何值），当 VDD 跌落到低于 VBOR （参数 D005）值的时间大于 TBOR（参数 35）就会复位器件。如果 VDD 跌落到 VBOR 以下的时间小于 TBOR 可能就不会发生复位。发生欠压复位以后，芯片将保持这种状态，直至VDD上升到VBOR 以上。

如果使能了上电延迟定时器，则它将在 VDD 上升到超过 VBOR 之后开始工作，并使芯片在延时 TPWRT （参数 33）期间保持复位。如果在上电延迟定时器运行过程中，VDD 电压降到了 VBOR 以下，芯片将重新回到欠压复位状态并且上电延迟定时器会恢复为初始状态。一旦 VDD 电压上升到 VBOR 以上，上电延迟定时器将执行一段额外的延时。

BOR 和上电延迟定时器 （PWRT）是分别配置的。使能 BOR 复位并不会自动使能 PWRT。

4.4.1 用软件使能 BOR

当 BOREN1:BOREN0 = 01 时，用户可以用软件使能或禁止 BOR。这可通过控制位 SBOREN（RCON<6>）完成。如前所述，将 SBOREN 置 1 可使能 BOR。清零 SBOREN 将完全禁止 BOR。SBOREN 位只工作在这个模式，其他情况下它将读为 0。

用软件控制 BOR 位可使用户能更灵活地定制应用程序以使其适应环境，而无需通过对器件重新编程来更改 BOR 配置。它还允许用户通过减少 BOR 消耗的电流，用软件调节器件的功耗。虽然 BOR 的电流通常很小，但是它可能对低功耗应用有一些影响。

注：	即使当 BOR 受软件控制时， BOR 复位电平仍将由 BORV1:BORV0 配置位设置。该值不能用软件更改。

4.4.2 检测 BOR

使能 BOR 后，当发生 BOR 或 POR 事件时，BOR 位总是复位为“0”。因此只通过读 BOR 位的状态很难确定是否发生过 BOR 事件。更可靠的方法是同时检查 POR 和 BOR 的状态。假定在发生任何 POR 事件后， POR 位被立即用软件复位为 1。如果 BOR 为 0 同时 POR 为 1，那么就可以断定已经发生了 BOR 事件。

4.4.3 在休眠模式下禁止 BOR

当 BOREN1:BOREN0 = 10 时，BOR 保持受硬件控制状态并且像前面描述的那样工作。每当器件进入休眠模式时，就会自动禁止BOR。当器件返回到任何其他工作模式时，又将自动重新使能 BOR。

此模式使应用程序能在有效执行代码的同时从欠压状态恢复，这也是器件最需要 BOR 保护的状况。同时，通过消除增加的 BOR 电流，可以省去休眠模式下的额外功耗。

表 4-1： BOR 配置

BOR 配置		SBOREN（RCON<6>）的状态	BOR 操作
BOREN1	BOREN0		
0	0	不可用	禁止 BOR ；必须对配置位重新编程才能使能 BOR。
0	1	可用	用软件使能 BOR ；工作模式由 SBOREN 控制。
1	0	不可用	用硬件使能 BOR ；在运行和空闲模式下有效，在休眠模式下禁止。
1	1	不可用	用硬件使能 BOR ；必须对配置位重新编程才能禁止 BOR。

4.5 器件复位定时器

PIC18F6390/6490/8390/8490 器件包含了三个独立的片上定时器，有助于调节上电复位过程。它们的主要功能是确保在代码执行之前器件时钟稳定。这些定时器是：

- 上电延迟定时器 （PWRT）
- 振荡器起振定时器 （OST）
- PLL 锁定延时

4.5.1 上电延迟定时器 （PWRT）

PIC18F6390/6490/8390/8490 器件的上电延迟定时器（PWRT）是一个 11 位计数器，使用 INTRC 时钟源作为时钟输入。该定时器可产生大约 2048 × 32 µs = 65.6 ms 的时间间隔。PWRT 计数期间，器件保持在复位状态。

上电延迟时间取决于 INTRC 时钟，并且由于温度和工艺的不同，不同器件的延迟时间也将各不相同。详情请参见 DC 参数 33。

通过清零配置位 $\overline{\text{PWRTEN}}$ 可使能 PWRT。

4.5.2 振荡器起振定时器 （OST）

在 PWRT 延时结束以后，由振荡器起振定时器（OST）提供一个 1024 振荡器周期的延时（从 OSC1 输入）（参数 33），从而确保晶振或谐振器的起振和稳定工作。

只有在 XT、LP、HS 和 HSPLL 模式下，并且仅当发生上电复位或从大多数功耗管理模式退出时，才启动 OST 延时。

4.5.3 PLL 锁定延时

当在 PLL 模式下使能 PLL 时，上电复位后的延时时序与其他振荡器模式略有不同。在 PLL 模式下需要使用一个独立的定时器来提供一段足够让 PLL 锁定主振荡器频率的固定延时。 PLL 锁定延时 （T_{PLL}）通常为 2 ms，且在振荡器起振延时后发生。

4.5.4 延时时序

上电延时时序如下：

1. POR 脉冲清零后，启动 PWRT 延时 （如果使能）。
2. 然后， OST 被激活。

总延迟时间将取决于振荡器的配置和 PWRT 的状态。图 4-3、图 4-4、图 4-5、图 4-6和图 4-7各自描述了不同的上电延时时序，其中上电延迟定时器均被使能，并且器件工作在 HS 振荡器模式下。图 4-3 到 4-6 也适用于在 XT 或 LP 模式下工作的器件。对于工作在 RC 模式下的器件以及禁止了 PWRT 的器件，将根本没有延时。

由于延时是由 POR 脉冲触发的，因此如果 $\overline{\text{MCLR}}$ 保持足够长时间的低电平，所有延时都将结束。将 $\overline{\text{MCLR}}$ 电平拉高后器件将立即开始执行代码 （图 4-5）。这对于测试或同步多个并行工作的 PIC18FXXXX 器件是非常有用的。

表 4-2： 不同情况下的延时

振荡器配置	上电复位 (2) 和欠压复位		从功耗管理模式退出
	$\overline{\text{PWRTEN}}$ = 0	$\overline{\text{PWRTEN}}$ = 1	
HSPLL	66 ms(1) + 1024 TOSC + 2 ms(2)	1024 TOSC + 2 ms(2)	1024 TOSC + 2 ms(2)
HS， XT， LP	66 ms(1) + 1024 TOSC	1024 TOSC	1024 TOSC
EC， ECIO	66 ms(1)	—	—
RC， RCIO	66 ms(1)	—	—
INTIO1， INTIO2	66 ms(1)	—	—

注 1：66 ms （65.5 ms）是上电延迟定时器 （PWRT）延迟时间的标称值。
2：2 ms 是 PLL 锁定所需的标称时间。

图 4-3： 上电延时时序 （MCLR 连接到 VDD， VDD 电压上升时间 < TPWRT）

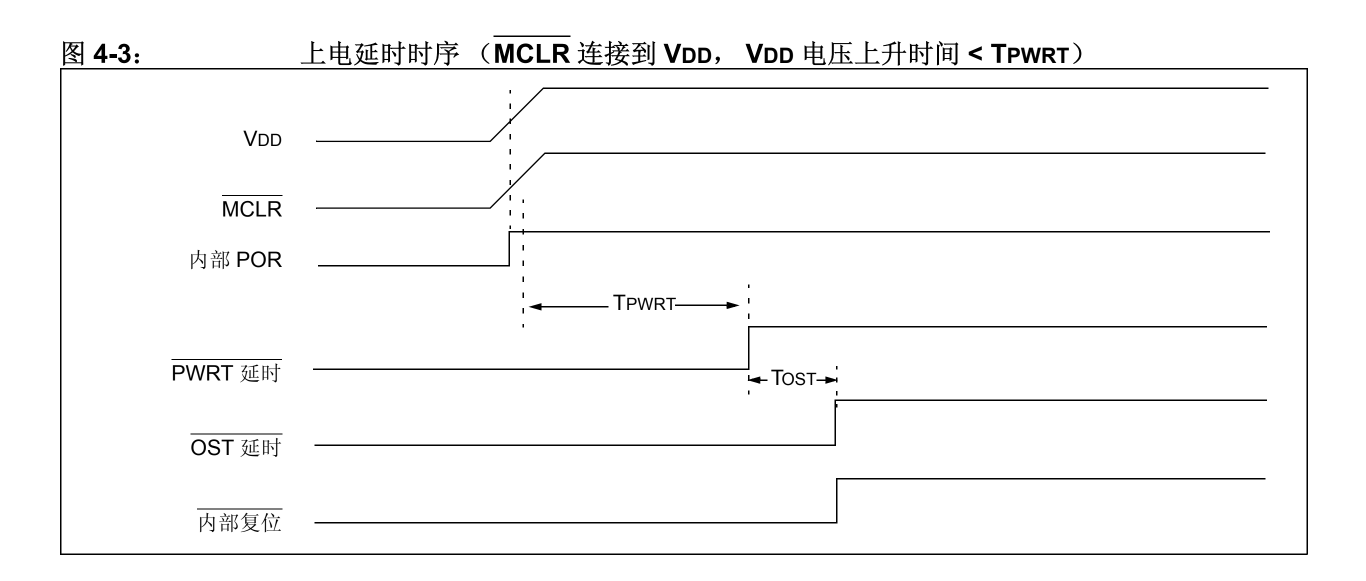


图 4-4： 上电延时时序 （MCLR 未连接到 VDD）：情形 1

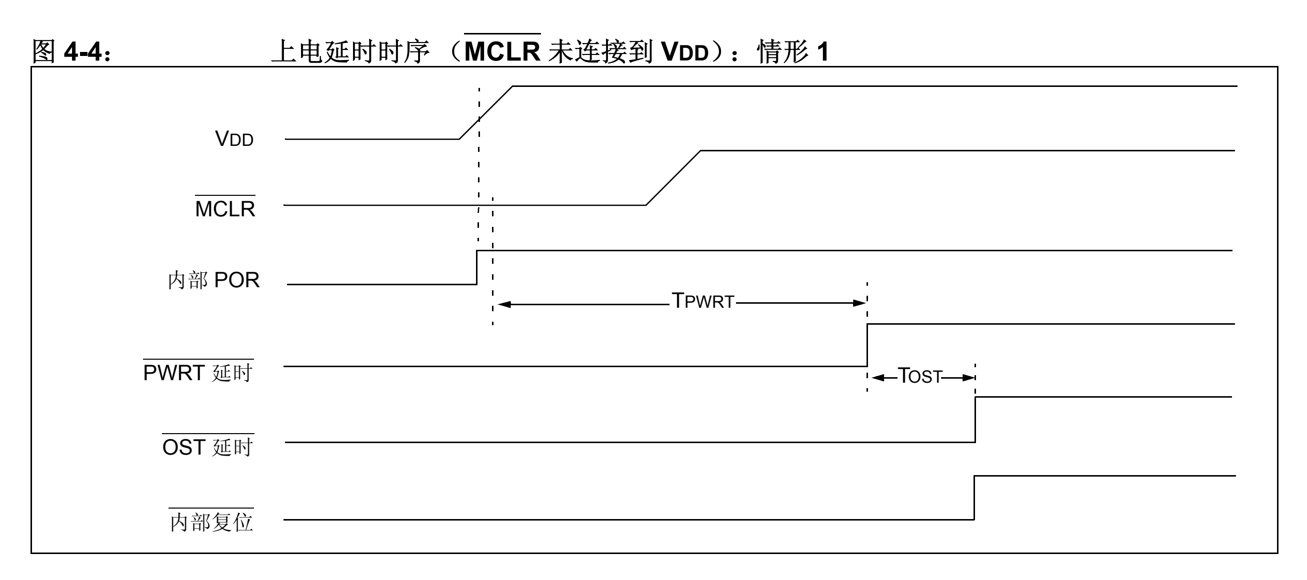


图 4-5： 上电延时时序 （MCLR 未连接到 VDD）：情形 2

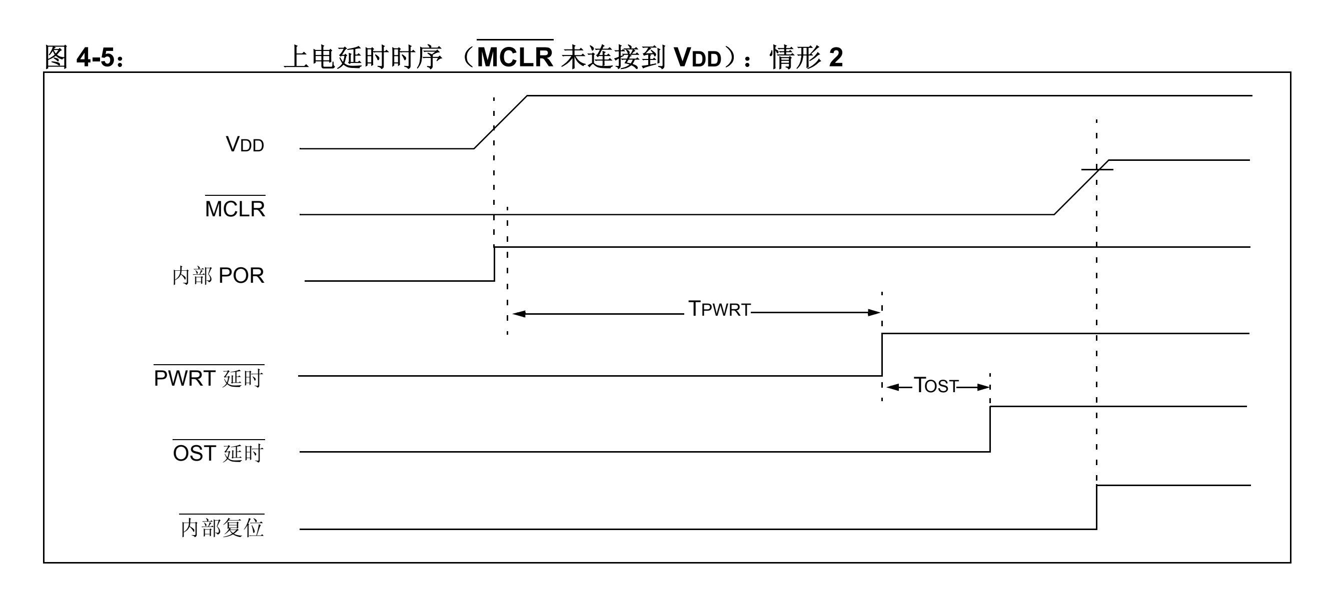


图 4-6： 缓慢上升时间 （$\overline{MCLR}$ 连接到 VDD， VDD 电压上升时间 > TPWRT）

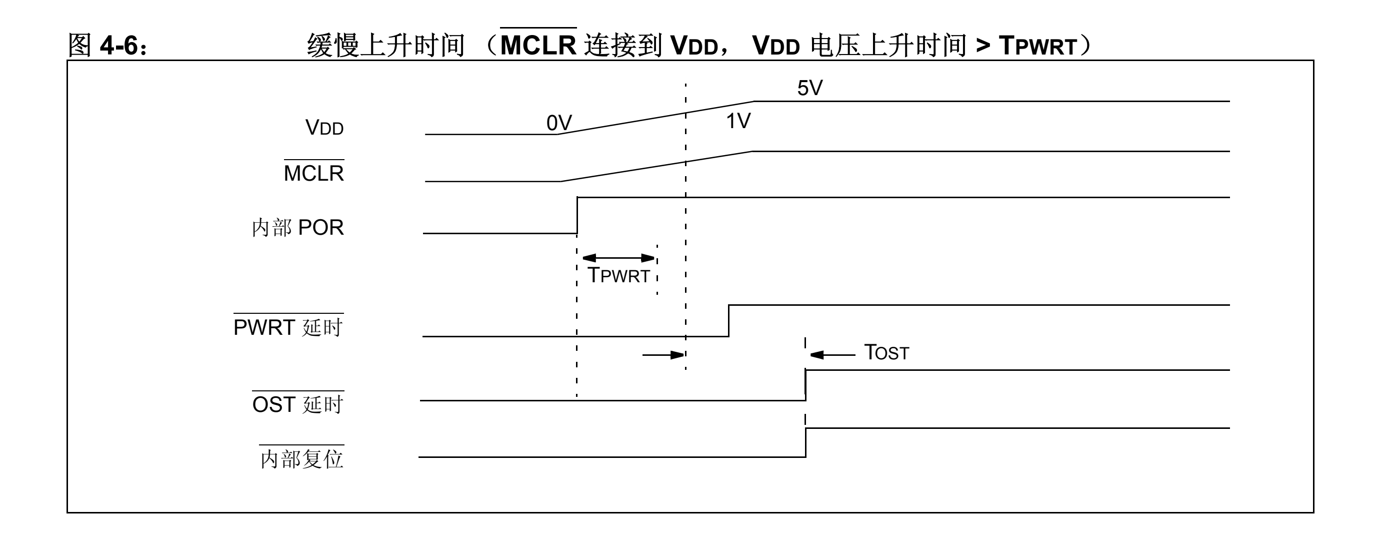
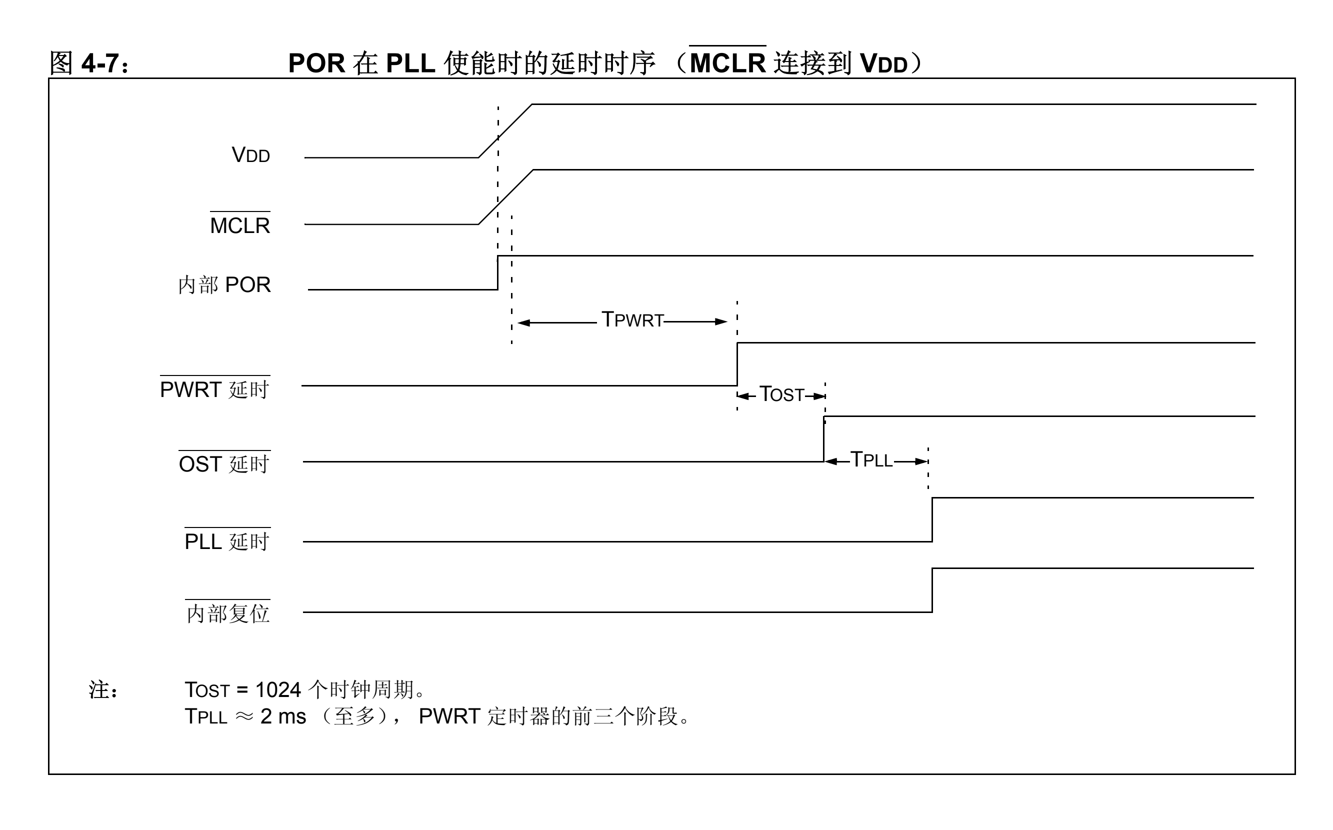


图 4-7： POR 在 PLL 使能时的延时时序 （$\overline{MCLR}$ 连接到 VDD）



注： TOST = 1024 个时钟周期。
TPLL ≈ 2 ms （至多）， PWRT 定时器的前三个阶段。

4.6 寄存器的复位状态

大多数寄存器不受复位的影响。在 POR 时这些寄存器的状态不确定，而在其他复位时它们的状态不变。而剩余寄存器则根据不同的复位类型被强制为 “复位状态”。

大多数寄存器不受 WDT 唤醒的影响，这是因为 WDT 唤醒被视为对正常工作的恢复。如表 4-3 所示，RCON 寄存器中的状态位：RI、TO、PD、POR 和 BOR，在不同的复位情形中会分别被置位或清零。可在软件中使用这些状态位判断复位的性质。

表 4-4 描述了所有特殊功能寄存器的复位状态。可以将这些复位状态分类为上电和欠压复位、主清零、WDT 复位以及 WDT 唤醒。

表 4-3： RCON 寄存器的状态位、含义以及初始化状态

条件	程序计数器	RCON 寄存器						STKPTR 寄存器	
		SBOREN	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$	STKFUL	STKUNF
上电复位	0000h	1	1	1	1	0	0	0	0
RESET 指令	0000h	u(2)	0	u	u	u	u	u	u
欠压复位	0000h	u(2)	1	1	1	u	0	u	u
功耗管理运行模式下的 $\overline{MCLR}$ 复位	0000h	u(2)	u	1	u	u	u	u	u
功耗管理空闲和休眠模式下的 $\overline{MCLR}$ 复位	0000h	u(2)	u	1	0	u	u	u	u
全功耗或功耗管理运行模式期间的 WDT 超时	0000h	u(2)	u	0	u	u	u	u	u
全功耗执行期间的 $\overline{MCLR}$ 复位	0000h	u(2)	u	u	u	u	u	u	u
堆栈满复位 （STVREN = 1）	0000h	u(2)	u	u	u	u	u	1	u
堆栈下溢复位 （STVREN = 1）	0000h	u(2)	u	u	u	u	u	u	1
堆栈下溢错误 （不是真正的复位， STVREN = 0）	0000h	u(2)	u	u	u	u	u	u	1
功耗管理空闲或休眠模式期间的 WDT 超时	PC + 2(1)	u(2)	u	0	0	u	u	u	u
通过中断从功耗管理模式退出	PC + 2(1)	u(2)	u	u	0	u	u	u	u

图注： u = 不变

注 1： 当器件被中断唤醒且 GIEH 或 GIEL 置 1 时， PC 装入中断矢量 （008h 或 0018h）。

2： 当软件使能 BOR 时（BOREN1:BOREN0 配置位 = 01 且 SBOREN = 1）， POR 位的复位状态是 1 且其他复位不能改变该状态。否则，其复位状态是 0。

表 **4-4**： 所有寄存器的初始化状态

寄存器	适用器件		上电复位， 欠压复位	MCLR 复位 WDT 复位 RESET 指令 堆栈复位	通过 WDT 或中断唤醒器件
TOSU	6X90	8X90	---0 0000	---0 0000	---0 uuuu(3)
TOSH	6X90	8X90	0000 0000	0000 0000	uuuu uuuu(3)
TOSL	6X90	8X90	0000 0000	0000 0000	uuuu uuuu(3)
STKPTR	6X90	8X90	uu-0 0000	00-0 0000	uu-u uuuu(3)
PCLATU	6X90	8X90	---0 0000	---0 0000	---u uuuu
PCLATH	6X90	8X90	0000 0000	0000 0000	uuuu uuuu
PCL	6X90	8X90	0000 0000	0000 0000	PC + 2(2)
TBLPTRU	6X90	8X90	--00 0000	--00 0000	--uu uuuu
TBLPTRH	6X90	8X90	0000 0000	0000 0000	uuuu uuuu
TBLPTRL	6X90	8X90	0000 0000	0000 0000	uuuu uuuu
TABLAT	6X90	8X90	0000 0000	0000 0000	uuuu uuuu
PRODH	6X90	8X90	xxxx xxxx	uuuu uuuu	uuuu uuuu
PRODL	6X90	8X90	xxxx xxxx	uuuu uuuu	uuuu uuuu
INTCON	6X90	8X90	0000 000x	0000 000u	uuuu uuuu(1)
INTCON2	6X90	8X90	1111 1111	1111 1111	uuuu uuuu(1)
INTCON3	6X90	8X90	1100 0000	1100 0000	uuuu uuuu(1)
INDF0	6X90	8X90	N/A	N/A	N/A
POSTINC0	6X90	8X90	N/A	N/A	N/A
POSTDEC0	6X90	8X90	N/A	N/A	N/A
PREINC0	6X90	8X90	N/A	N/A	N/A
PLUSW0	6X90	8X90	N/A	N/A	N/A
FSR0H	6X90	8X90	---- xxxx	---- uuuu	---- uuuu
FSR0L	6X90	8X90	xxxx xxxx	uuuu uuuu	uuuu uuuu
WREG	6X90	8X90	xxxx xxxx	uuuu uuuu	uuuu uuuu
INDF1	6X90	8X90	N/A	N/A	N/A
POSTINC1	6X90	8X90	N/A	N/A	N/A
POSTDEC1	6X90	8X90	N/A	N/A	N/A
PREINC1	6X90	8X90	N/A	N/A	N/A
PLUSW1	6X90	8X90	N/A	N/A	N/A

图注： u = 不变， x = 未知， - = 未用 （读为 0）， q = 取值视情况而定。
阴影单元格表示不适用于指定器件。

注 **1**： INTCONx 或 PIRx 寄存器中的一位或多位会受到影响 （引起唤醒）。

2： 当器件被中断唤醒且 GIEL 或 GIEH 位被置 1 时， PC 装入中断矢量 （0008h 或 0018h）。

3： 当器件被中断唤醒且 GIEL 或 GIEH 位被置 1 时，由 PC 更新 TOSU、 TOSH 和 TOSL 的当前值。将 STKPTR 修改为指向硬件堆栈的下一个单元。

4： 具体条件下的复位值，请参见表 4-3。

5： 根据所选择的振荡器模式使能 PORTA、 LATA 和 TRISA 中的 bit 6 和 bit 7。若未将这两位配置为 PORTA 引脚，则它们将被禁止并读为 0。

6： 这些寄存器在发生 POR 时被清零，在发生 BOR 时保持不变。

表 4-4： 所有寄存器的初始化状态（续）

寄存器	适用器件		上电复位， 欠压复位	MCLR 复位 WDT 复位 RESET 指令 堆栈复位	通过 WDT 或中断唤醒器件
FSR1H	6X90	8X90	---- xxxx	---- uuuu	---- uuuu
FSR1L	6X90	8X90	xxxx xxxx	uuuu uuuu	uuuu uuuu
BSR	6X90	8X90	---- 0000	---- 0000	---- uuuu
INDF2	6X90	8X90	N/A	N/A	N/A
POSTINC2	6X90	8X90	N/A	N/A	N/A
POSTDEC2	6X90	8X90	N/A	N/A	N/A
PREINC2	6X90	8X90	N/A	N/A	N/A
PLUSW2	6X90	8X90	N/A	N/A	N/A
FSR2H	6X90	8X90	---- xxxx	---- uuuu	---- uuuu
FSR2L	6X90	8X90	xxxx xxxx	uuuu uuuu	uuuu uuuu
STATUS	6X90	8X90	---x xxxx	---u uuuu	---u uuuu
TMR0H	6X90	8X90	0000 0000	0000 0000	uuuu uuuu
TMR0L	6X90	8X90	xxxx xxxx	uuuu uuuu	uuuu uuuu
T0CON	6X90	8X90	1111 1111	1111 1111	uuuu uuuu
OSCCON	6X90	8X90	0100 q000	0100 00q0	uuuu uuqu
HLVDCON	6X90	8X90	0-00 0101	0-00 0101	u-uu uuuu
WDTCON	6X90	8X90	---- ---0	---- ---0	---- ---u
RCON[(4)]	6X90	8X90	0q-1 11q0	0q-q qquu	uq-u qquu
TMR1H	6X90	8X90	xxxx xxxx	uuuu uuuu	uuuu uuuu
TMR1L	6X90	8X90	xxxx xxxx	uuuu uuuu	uuuu uuuu
T1CON	6X90	8X90	0000 0000	u0uu uuuu	uuuu uuuu
TMR2	6X90	8X90	0000 0000	0000 0000	uuuu uuuu
PR2	6X90	8X90	1111 1111	1111 1111	1111 1111
T2CON	6X90	8X90	-000 0000	-000 0000	-uuu uuuu
SSPBUF	6X90	8X90	xxxx xxxx	uuuu uuuu	uuuu uuuu
SSPADD	6X90	8X90	0000 0000	0000 0000	uuuu uuuu
SSPSTAT	6X90	8X90	0000 0000	0000 0000	uuuu uuuu
SSPCON1	6X90	8X90	0000 0000	0000 0000	uuuu uuuu
SSPCON2	6X90	8X90	0000 0000	0000 0000	uuuu uuuu

图注： u = 不变， x = 未知， - = 未用 （读为 0）， q = 取值视情况而定。
阴影单元格表示不适用于指定器件。

注 1： INTCONx 或 PIRx 寄存器中的一位或多位会受到影响 （引起唤醒）。

2： 当器件被中断唤醒且 GIEL 或 GIEH 位被置 1 时， PC 装入中断矢量 （0008h 或 0018h）。

3： 当器件被中断唤醒且 GIEL 或 GIEH 位被置 1 时，由 PC 更新 TOSU、 TOSH 和 TOSL 的当前值。将 STKPTR 修改为指向硬件堆栈的下一个单元。

4： 具体条件下的复位值，请参见表 4-3。

5： 根据所选择的振荡器模式使能 PORTA、 LATA 和 TRISA 中的 bit 6 和 bit 7。若未将这两位配置为 PORTA 引脚，则它们将被禁止并读为 0。

6： 这些寄存器在发生 POR 时被清零，在发生 BOR 时保持不变。

表 4-4： 所有寄存器的初始化状态（续）

寄存器	适用器件		上电复位， 欠压复位	MCLR 复位 WDT 复位 RESET 指令 堆栈复位	通过 WDT 或中断唤醒器件
ADRESH	6X90	8X90	xxxx xxxx	uuuu uuuu	uuuu uuuu
ADRESL	6X90	8X90	xxxx xxxx	uuuu uuuu	uuuu uuuu
ADCON0	6X90	8X90	--00 0000	--00 0000	--uu uuuu
ADCON1	6X90	8X90	--00 0000	--00 0000	--uu uuuu
ADCON2	6X90	8X90	0-00 0000	0-00 0000	u-uu uuuu
CCPR1H	6X90	8X90	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR1L	6X90	8X90	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCP1CON	6X90	8X90	--00 0000	--00 0000	--uu uuuu
CCPR2H	6X90	8X90	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR2L	6X90	8X90	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCP2CON	6X90	8X90	--00 0000	--00 0000	--uu uuuu
CVRCON	6X90	8X90	000- 0000	000- 0000	uuu- uuuu
CMCON	6X90	8X90	0000 0111	0000 0111	uuuu uuuu
TMR3H	6X90	8X90	xxxx xxxx	uuuu uuuu	uuuu uuuu
TMR3L	6X90	8X90	xxxx xxxx	uuuu uuuu	uuuu uuuu
T3CON	6X90	8X90	0000 0000	uuuu uuuu	uuuu uuuu
SPBRG1	6X90	8X90	0000 0000	0000 0000	uuuu uuuu
RCREG1	6X90	8X90	0000 0000	0000 0000	uuuu uuuu
TXREG1	6X90	8X90	0000 0000	0000 0000	uuuu uuuu
TXSTA1	6X90	8X90	0000 0010	0000 0010	uuuu uuuu
RCSTA1	6X90	8X90	0000 000x	0000 000x	uuuu uuuu
IPR3	6X90	8X90	-111 ----	-111 ----	-uuu ----
PIR3	6X90	8X90	-000 ----	-000 ----	-uuu ----(1)
PIE3	6X90	8X90	-000 ----	-000 ----	-uuu ----
IPR2	6X90	8X90	11-- 1111	11-- 1111	uu-- uuuu
PIR2	6X90	8X90	00-- 0000	00-- 0000	uu-- uuuu(1)
PIE2	6X90	8X90	00-- 0000	00-- 0000	uu-- uuuu
IPR1	6X90	8X90	-111 1111	-111 1111	-uuu uuuu
PIR1	6X90	8X90	-000 0000	-000 0000	-uuu uuuu(1)
PIE1	6X90	8X90	-000 0000	-000 0000	-uuu uuuu
OSCTUNE	6X90	8X90	00-0 0000	00-0 0000	uu-u uuuu

图注： u = 不变， x = 未知， - = 未用 （读为 0）， q = 取值视情况而定。
阴影单元格表示不适用于指定器件。

注 1： INTCONx 或 PIRx 寄存器中的一位或多位会受到影响 （引起唤醒）。
2： 当器件被中断唤醒且 GIEL 或 GIEH 位被置 1 时， PC 装入中断矢量 （0008h 或 0018h）。
3： 当器件被中断唤醒且 GIEL 或 GIEH 位被置 1 时，由 PC 更新 TOSU、 TOSH 和 TOSL 的当前值。将 STKPTR 修改为指向硬件堆栈的下一个单元。
4： 具体条件下的复位值，请参见表 4-3。
5： 根据所选择的振荡器模式使能 PORTA、 LATA 和 TRISA 中的 bit 6 和 bit 7。若未将这两位配置为 PORTA 引脚，则它们将被禁止并读为 0。
6： 这些寄存器在发生 POR 时被清零，在发生 BOR 时保持不变。

表 4-4：　所有寄存器的初始化状态（续）

寄存器	适用器件		上电复位，欠压复位	MCLR 复位 WDT 复位 RESET 指令 堆栈复位	通过 WDT 或中断唤醒器件
TRISJ	6X90	8X90	1111 1111	1111 1111	uuuu uuuu
TRISH	6X90	8X90	1111 1111	1111 1111	uuuu uuuu
TRISG	6X90	8X90	---1 1111	---1 1111	---u uuuu
TRISF	6X90	8X90	1111 1111	1111 1111	uuuu uuuu
TRISE	6X90	8X90	1111 ----	1111 ----	uuuu ----
TRISD	6X90	8X90	1111 1111	1111 1111	uuuu uuuu
TRISC	6X90	8X90	1111 1111	1111 1111	uuuu uuuu
TRISB	6X90	8X90	1111 1111	1111 1111	uuuu uuuu
TRISA(5)	6X90	8X90	1111 1111(5)	1111 1111(5)	uuuu uuuu(5)
LATJ	6X90	8X90	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATH	6X90	8X90	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATG	6X90	8X90	---x xxxx	---u uuuu	---u uuuu
LATF	6X90	8X90	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATE	6X90	8X90	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATD	6X90	8X90	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATC	6X90	8X90	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATB	6X90	8X90	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATA(5)	6X90	8X90	xxxx xxxx(5)	uuuu uuuu(5)	uuuu uuuu(5)
PORTJ	6X90	8X90	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTH	6X90	8X90	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTG	6X90	8X90	--xx xxxx	--uu uuuu	--uu uuuu
PORTF	6X90	8X90	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTE	6X90	8X90	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTD	6X90	8X90	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTC	6X90	8X90	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTB	6X90	8X90	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTA(5)	6X90	8X90	xx0x 0000(5)	uu0u 0000(5)	uuuu uuuu(5)
SPBRGH1	6X90	8X90	0000 0000	0000 0000	uuuu uuuu
BAUDCON1	6X90	8X90	01-0 0-00	01-0 0-00	uu-u u-uu

图注：　u = 不变， x = 未知， - = 未用 （读为 0）， q = 取值视情况而定。
阴影单元格表示不适用于指定器件。

注　1： INTCONx 或 PIRx 寄存器中的一位或多位会受到影响 （引起唤醒）。
2： 当器件被中断唤醒且 GIEL 或 GIEH 位被置 1 时， PC 装入中断矢量 （0008h 或 0018h）。
3： 当器件被中断唤醒且 GIEL 或 GIEH 位被置 1 时，由 PC 更新 TOSU、 TOSH 和 TOSL 的当前值。将 STKPTR 修改为指向硬件堆栈的下一个单元。
4： 具体条件下的复位值，请参见表 4-3。
5： 根据所选择的振荡器模式使能 PORTA、 LATA 和 TRISA 中的 bit 6 和 bit 7。若未将这两位配置为 PORTA 引脚，则它们将被禁止并读为 0。
6： 这些寄存器在发生 POR 时被清零，在发生 BOR 时保持不变。

表 4-4： 所有寄存器的初始化状态（续）

寄存器	适用器件		上电复位，欠压复位	MCLR 复位 WDT 复位 RESET 指令 堆栈复位	通过 WDT 或中断唤醒器件
LCDDATA23	6X90	8X90	0000 0000	0000 0000	uuuu uuuu
LCDDATA22	6X90	8X90	0000 0000	0000 0000	uuuu uuuu
LCDDATA21	6X90	8X90	0000 0000	0000 0000	uuuu uuuu
LCDDATA20	6X90	8X90	0000 0000	0000 0000	uuuu uuuu
LCDDATA19	6X90	8X90	0000 0000	0000 0000	uuuu uuuu
LCDDATA18	6X90	8X90	0000 0000	0000 0000	uuuu uuuu
LCDDATA17	6X90	8X90	0000 0000	0000 0000	uuuu uuuu
LCDDATA16	6X90	8X90	0000 0000	0000 0000	uuuu uuuu
LCDDATA15	6X90	8X90	0000 0000	0000 0000	uuuu uuuu
LCDDATA14	6X90	8X90	0000 0000	0000 0000	uuuu uuuu
LCDDATA13	6X90	8X90	0000 0000	0000 0000	uuuu uuuu
LCDDATA12	6X90	8X90	0000 0000	0000 0000	uuuu uuuu
LCDDATA11	6X90	8X90	0000 0000	0000 0000	uuuu uuuu
SPBRG2	6X90	8X90	0000 0000	0000 0000	uuuu uuuu
RCREG2	6X90	8X90	0000 0000	0000 0000	uuuu uuuu
TXREG2	6X90	8X90	0000 0000	0000 0000	uuuu uuuu
TXSTA2	6X90	8X90	0000 0010	0000 0010	uuuu uuuu
RCSTA2	6X90	8X90	0000 000x	0000 000x	uuuu uuuu
LCDDATA10	6X90	8X90	0000 0000	0000 0000	uuuu uuuu
LCDDATA9	6X90	8X90	0000 0000	0000 0000	uuuu uuuu
LCDDATA8	6X90	8X90	0000 0000	0000 0000	uuuu uuuu
LCDDATA7	6X90	8X90	0000 0000	0000 0000	uuuu uuuu
LCDDATA6	6X90	8X90	0000 0000	0000 0000	uuuu uuuu
LCDDATA5	6X90	8X90	0000 0000	0000 0000	uuuu uuuu
LCDDATA4	6X90	8X90	0000 0000	0000 0000	uuuu uuuu
LCDDATA3	6X90	8X90	0000 0000	0000 0000	uuuu uuuu
LCDDATA2	6X90	8X90	0000 0000	0000 0000	uuuu uuuu
LCDDATA1	6X90	8X90	0000 0000	0000 0000	uuuu uuuu
LCDDATA0	6X90	8X90	0000 0000	0000 0000	uuuu uuuu

图注： u = 不变， x = 未知， - = 未用 （读为 0）， q = 取值视情况而定。
阴影单元格表示不适用于指定器件。

注 1： INTCONx 或 PIRx 寄存器中的一位或多位会受到影响 （引起唤醒）。
2： 当器件被中断唤醒且 GIEL 或 GIEH 位被置 1 时， PC 装入中断矢量 （0008h 或 0018h）。
3： 当器件被中断唤醒且 GIEL 或 GIEH 位被置 1 时，由 PC 更新 TOSU、 TOSH 和 TOSL 的当前值。将 STKPTR 修改为指向硬件堆栈的下一个单元。
4： 具体条件下的复位值，请参见表 4-3。
5： 根据所选择的振荡器模式使能 PORTA、 LATA 和 TRISA 中的 bit 6 和 bit 7。若未将这两位配置为 PORTA 引脚，则它们将被禁止并读为 0。
6： 这些寄存器在发生 POR 时被清零，在发生 BOR 时保持不变。

表 4-4：　　所有寄存器的初始化状态（续）

寄存器	适用器件		上电复位， 欠压复位	MCLR 复位 WDT 复位 RESET 指令 堆栈复位	通过 WDT 或中断唤醒器件
LCDSE5	6X90	8X90	0000 0000[(6)]	0000 0000	uuuu uuuu
LCDSE4	6X90	8X90	0000 0000[(6)]	0000 0000	uuuu uuuu
LCDSE3	6X90	8X90	0000 0000[(6)]	0000 0000	uuuu uuuu
LCDSE2	6X90	8X90	0000 0000[(6)]	0000 0000	uuuu uuuu
LCDSE1	6X90	8X90	0000 0000[(6)]	0000 0000	uuuu uuuu
LCDSE0	6X90	8X90	0000 0000[(6)]	0000 0000	uuuu uuuu
LCDCON	6X90	8X90	000- 0000	000- 0000	uuu- uuuu
LCDPS	6X90	8X90	0000 0000	0000 0000	uuuu uuuu

图注：　u = 不变， x = 未知， - = 未用 （读为 0）， q = 取值视情况而定。
阴影单元格表示不适用于指定器件。

注　**1：** INTCONx 或 PIRx 寄存器中的一位或多位会受到影响 （引起唤醒）。

2： 当器件被中断唤醒且 GIEL 或 GIEH 位被置 1 时， PC 装入中断矢量 （0008h 或 0018h）。

3： 当器件被中断唤醒且 GIEL 或 GIEH 位被置 1 时，由 PC 更新 TOSU、 TOSH 和 TOSL 的当前值。将 STKPTR 修改为指向硬件堆栈的下一个单元。

4： 具体条件下的复位值，请参见表 4-3。

5： 根据所选择的振荡器模式使能 PORTA、 LATA 和 TRISA 中的 bit 6 和 bit 7。若未将这两位配置为 PORTA 引脚，则它们将被禁止并读为 0。

6： 这些寄存器在发生 POR 时被清零，在发生 BOR 时保持不变。

5.0 存储器构成

PIC18 闪存单片机有两种类型的存储器：

- 程序存储器
- 数据 RAM

在哈佛架构的器件中，数据和程序存储器使用不同的总线，因而可同时访问这两种存储器空间。

第 6.0 节 “闪存程序存储器”提供了关于闪存程序存储器操作的更多详细信息。

5.1 程序存储器构成

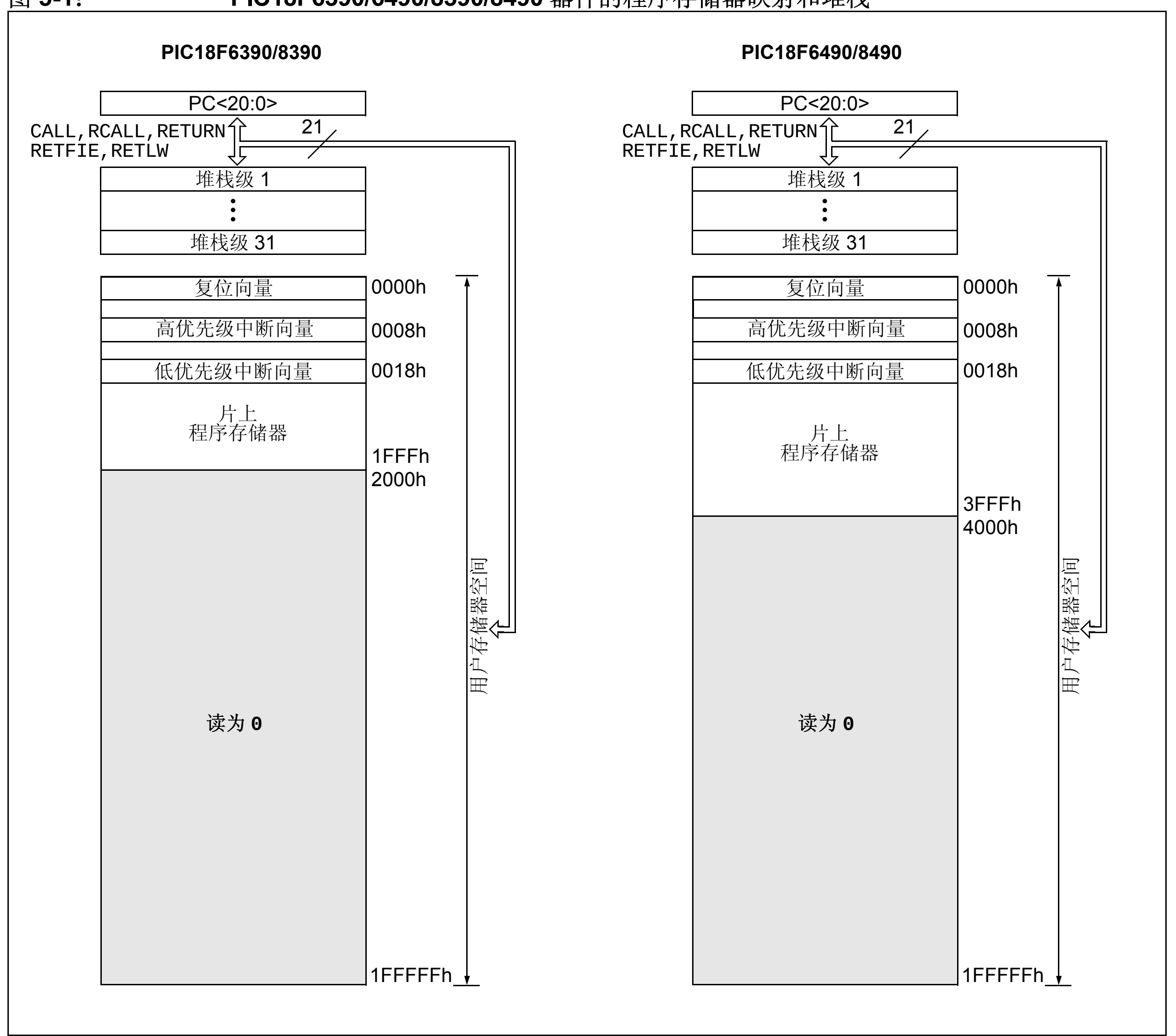
PIC18 单片机具有 21 位程序计数器，可以对 2 MB 的程序存储器空间进行寻址。访问存储器物理地址上边界和这个2 MB地址之间的存储单元会返回全0（NOP指令）。

PIC18FX390 有 8 KB 的闪存存储器并且可以存储最多 4,096 个单字指令； PIC18FX490 具有 16 KB 的闪存存储器并且可以存储最多 8,192 个单字指令。

PIC18 器件有两个中断向量。复位向量地址为 0000h，中断向量地址为 0008h 和 0018h。

图 5-1 显示了 PIC18F6390/6490/8390/8490 器件的程序存储器映射。

图 5-1： PIC18F6390/6490/8390/8490 器件的程序存储器映射和堆栈



5.1.1 程序计数器

程序计数器（Program Counter，PC）指定要取出执行的指令的地址。PC 中的数为 21 位二进制格式，且保存在三个不同的 8 位寄存器中。存储低字节的寄存器称为 PCL 寄存器，该寄存器可读写。存储高字节的寄存器，即 PCH 寄存器，存储 PC<15:8> 位，该寄存器不可直接读写。更新 PCH 寄存器的操作是通过更新 PCLATH 寄存器实现的。存储最高字节的寄存器称为 PCU。该寄存器存储 PC<20:16> 位，它也不能直接读写。通过 PCLATU 寄存器更新 PCU 寄存器。

PCLATH 和 PCLATU 的内容通过执行任何写 PCL 的操作传送到程序计数器。同样，程序计数器的两个高字节通过读 PCL 的操作被传送到 PCLATH 和 PCLATU。这对于计算 PC 的偏移量很有用处（见第 **5.1.4.1 节“计算 GOTO”**）。

PC 是按字节寻址程序存储器的。为了防止 PC 不能正确获取字指令，需要将 PCL 的最低有效位固定取值为 0。 PC 每次加 2 来寻址程序存储器中的顺序指令。

CALL、RCALL、GOTO 和程序转移指令直接写入程序计数器。对于这些指令，PCLATH 和 PCLATU 的内容不会传送到程序计数器。

5.1.2 返回地址堆栈

返回地址堆栈允许保存最多 31 个程序调用地址和中断向量。当执行 CALL、RCALL 指令或响应中断时，PC 的值会被压入该堆栈。而执行 RETURN、RETLW 或 RETFIE 指令时，PC 值会从堆栈弹出。PCLATU 和 PCLATH 不受 RETURN 或 CALL 指令的影响。

通过 21 位的 RAM 和一个 5 位的堆栈指针来实现 31 级的堆栈操作。堆栈既不占用程序存储器空间也不占用数据存储器空间。堆栈指针是可读写的，并且通过栈顶的特殊文件寄存器可以读写栈顶地址。也可以使用这些寄存器将数据压入堆栈或者从堆栈弹出。

执行 CALL 类型的指令时，产生进栈操作：首先堆栈指针加 1，并且将 PC 的内容写入堆栈指针所指向的地址单元（PC 已经指向 CALL 下一条指令）。执行 RETURN 类型的指令时，产生出栈操作：STKPTR 寄存器所指向的地址单元的内容会被传递给 PC，然后堆栈指针减 1。

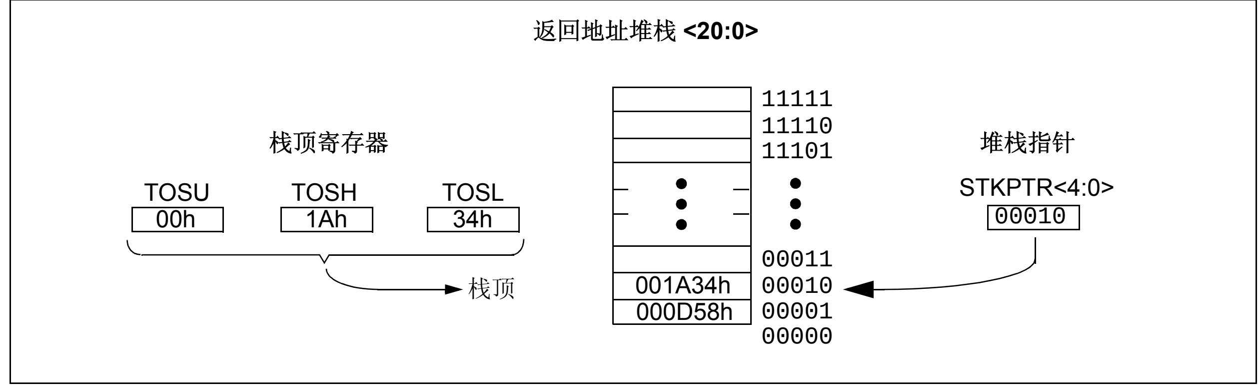
所有复位后，堆栈指针均会初始化为 00000。堆栈指针值 00000 不指向任何 RAM 单元，它仅仅是一个复位值。状态位表明堆栈是否已满，是上溢还是下溢。

5.1.2.1 访问栈顶

只可读写返回地址堆栈的栈顶（Top-of-Stack，TOS）。有三个寄存器 TOSU:TOSH:TOSL 用于保存由 STKPTR 寄存器的低 5 位所指向的堆栈单元的内容（图 5-2）。这可以让用户在必要时实现软件堆栈。在 CALL、RCALL 或中断后，软件可以通过读取 TOSU:TOSH:TOSL 寄存器来读取压入堆栈的值。这些值可以被置入由用户定义的软件堆栈。返回时，软件将这些值存回 TOSU:TOSH:TOSL 并执行返回。

为防止意外的堆栈操作，访问堆栈时用户必须禁止全局中断使能位。


图 5-2： 返回地址堆栈和相关寄存器



 初稿

5.1.2.2 返回堆栈指针（STKPTR）

STKPTR 寄存器（寄存器 5-1）包含堆栈指针值、STKFUL（堆栈满）状态位和 STKUNF（堆栈下溢）状态位。堆栈指针值可为 0 到 31 范围内的整数。向堆栈压入值前，堆栈指针加 1；而从堆栈弹出值后，堆栈指针减 1。复位时，堆栈指针值为零。用户可以读写堆栈指针的值。实时操作系统可以利用此特性对返回堆栈进行维护。

当向堆栈压入 PC 值 31 次（且没有值从堆栈弹出）后，STKFUL 位就会置 1。通过软件或 POR 使 STKFUL 位清零。

由 STVREN（堆栈溢出复位使能）配置位的状态决定堆栈满时将执行的操作（有关器件配置位的介绍，请参见**第 23.1 节 "配置位"**）。如果 STVREN 位已经置 1（默认），第 31 次进栈将把（PC+2）值压入堆栈，从而将 STKFUL 位置 1，并复位器件。STKFUL 位将保持置 1，而堆栈指针将被清零。

如果将 STVREN 位清零，第 31 次进栈时 STKFUL 位将会置 1，堆栈指针加 1 变为 31。任何其他进栈操作都不会覆盖第 31 次进栈的值，并且 STKPTR 将保持为 31。

当堆栈弹出次数足够卸空堆栈时，下一次出栈会向 PC 返回一个零值，并将 STKUNF 位置 1，而堆栈指针则保持为 0。STKUNF 位将保持置 1，直到软件清零或发生 POR。

注：	下溢时，将零值返回给 PC，会使程序指向复位向量，此时可以验证堆栈状态并采取相应的操作。这与复位不同，因为下溢时 SFR 的内容不受影响。

5.1.2.3 PUSH 和 POP 指令

由于栈顶（TOS）是可以读写的，因此将值压入堆栈或从堆栈弹出而不影响程序的正常执行是非常理想的。PIC18 指令集包含两个用于堆栈操作的指令 PUSH 和 POP，使用这两个指令可在软件控制下对 TOS 执行操作。然后就可以修改 TOSU、TOSH 和 TOSL，将数据或返回地址压入堆栈。

PUSH 指令将当前的 PC 值压入堆栈。堆栈指针加 1 并将当前的 PC 值装入堆栈。

POP 指令通过将堆栈指针减 1 来丢掉当前的 TOS 值。然后前一个入栈的值就成为了 TOS 值。

寄存器 5-1： STKPTR：堆栈指针寄存器

R/C-0	R/C-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
STKFUL[(1)]	STKUNF[(1)]	—	SP4	SP3	SP2	SP1	SP0
bit 7							bit 0

bit 7 **STKFUL**：堆栈满标志位 [(1)]
1 = 堆栈满或溢出
0 = 堆栈未满或未溢出

bit 6[(1)] **STKUNF**：堆栈下溢标志位 [(1)]
1 = 发生堆栈下溢
0 = 未发生堆栈下溢

bit 5 **未用位**：读为 0

bit 4-0 **SP4:SP0**：堆栈指针地址位

注 **1**：通过用户软件或 POR 清零 bit 7 和 bit 6。

图注：			
R = 可读位	W = 可写位	U = 未用位	C = 只可清零位
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

5.1.2.4 堆栈满和下溢复位

通过将配置寄存器 4L 中的 STVREN 位置 1，来使能在堆栈溢出或下溢时的器件复位。当 STVREN 位置 1 时，堆栈满或堆栈下溢状态会将相应的STKFUL或STKUNF位置 1，然后使器件复位。当 STVREN 位清零时，堆栈满或堆栈下溢状态会将相应的 STKFUL 或 STKUNF 位置 1，但不会使器件复位。只能通过用户软件或上电复位使 STKFUL 或 STKUNF 位清零。

5.1.3 快速寄存器堆栈

为 Status、WREG 和 BSR 寄存器提供的快速寄存器堆栈具有从中断“快速返回”的功能。此堆栈只有一级且不可读写。当处理器转入中断向量处执行时，它装入对应寄存器的当前值。所有中断源都会将值压入堆栈寄存器。如果使用 `RETFIE, FAST` 指令从中断返回，这些寄存器中的值就会重新装回原工作寄存器。

如果同时使能了低优先级中断和高优先级中断，从低优先级中断返回时，无法可靠地使用堆栈寄存器。如果在为低优先级中断提供服务时，发生了高优先级中断，则低优先级中断存储在堆栈寄存器中的值将被覆盖。在低优先级中断过程中，用户必须用软件保存关键寄存器的值。

如果未使用中断优先级，所有中断都可以使用快速寄存器堆栈从中断返回。如果没有使用中断，快速寄存器堆栈可以用于在子程序调用结束后恢复 Status、WREG 和 BSR 寄存器。要在子程序调用中使用快速寄存器堆栈，必须执行 `CALL label, FAST` 指令将 Status、WREG 和 BSR 寄存器的内容存入快速寄存器堆栈。在调用结束后执行 `RETURN, FAST` 指令，弹出快速寄存器堆栈中的值并恢复这些寄存器。

例 5-1给出了一个在子程序调用和返回期间使用快速寄存器堆栈的源代码示例。

例 5-1： 快速寄存器堆栈代码示例

```
CALL SUB1, FAST    ;STATUS, WREG, BSR
                   ;SAVED IN FAST REGISTER
                   ;STACK
        •
        •

SUB1    •
        •
   RETURN FAST     ;RESTORE VALUES SAVED
                   ;IN FAST REGISTER STACK
```

5.1.4 程序存储器中的查找表

有的编程需要在程序存储器中创建数据结构或查找表。对于 PIC18 器件而言，可以用两种方式实现查找表：

- 计算 GOTO
- 表读

5.1.4.1 计算 GOTO

计算 GOTO 是通过向程序计数器加一个偏移量来实现的。例 5-2 给出了一个示例。

使用 `ADDWF PCL` 指令和一组 `RETLW nn` 指令可以创建一个查找表。在调用该表前，会先将查找表中的偏移量装入 W 寄存器。被调用子程序的第一条指令是 `ADDWF PCL` 指令。接下去执行的一条是 `RETLW nn` 指令，它将向调用函数返回值 nn。

偏移量 （WREG 中的值）指定程序计数器应该增加的字节数，其值应当为 2 的倍数 （`LSb = 0`）。

在这种方式中，每个指令单元只能存储一个数据字节，并且要求返回地址堆栈还有空闲单元。

例 5-2： 使用偏移量计算 GOTO

```
        MOVF    OFFSET, W
        CALL    TABLE
ORG     nn00h
TABLE   ADDWF   PCL
        RETLW   nnh
        RETLW   nnh
        RETLW   nnh
        .
        .
        .
```

5.1.4.2 表读

有一种更好的方法可以将数据存储在程序存储器中，这种方法允许在每个指令单元存储 2 个字节的数据。

在编程时，每个程序字可以存储 2 个字节的查找表数据。表指针寄存器 （TBLPTR）指定字节地址，而表锁存器 （TABLAT）储存从程序存储器中读取的数据。一次只能从程序存储器读取一个字节。

第 **6.1** 节 **“表读操作”**将进一步讨论表读操作。

5.2 PIC18 指令周期

5.2.1 时钟分配

来自内部或外部时钟源的单片机时钟输入都将在内部被四分频以产生四个互不重叠的正交时钟信号（Q1、Q2、Q3 和 Q4）。程序计数器在每个 Q1 递增。在 Q4 期间，从程序存储器取指并将指令锁存到指令寄存器 （IR）中。指令的译码和执行在下一个 Q1 到 Q4 周期完成。图 5-3 所示为时钟和指令执行的流程图。

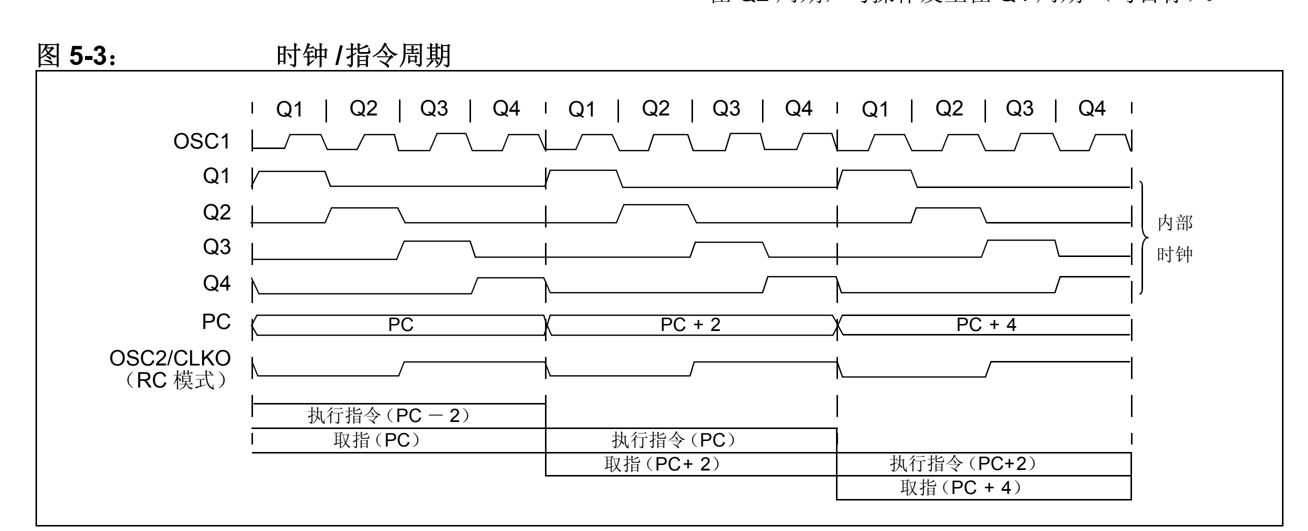
5.2.2 指令流 / 流水线

一个 “指令周期”由 4 个 Q 周期组成，即 Q1 到 Q4。取指和执行指令是流水执行的，用一个指令周期来取指，而用另一个指令周期译码和执行指令。但由于是流水线操作，所以每条指令的等效执行时间都是一个指令周期。如果某条指令改变了程序计数器 （如 GOTO 指令），则需要两个指令周期才能完成该指令 （见例 5-3）。

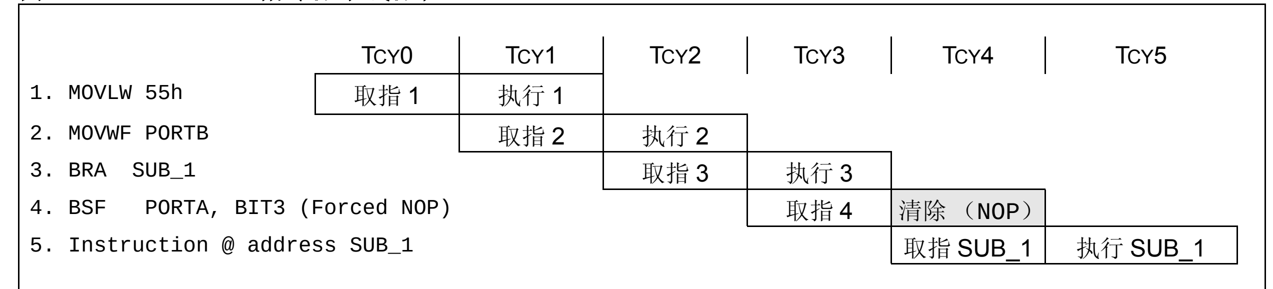
在 Q1 周期，程序计数器 （PC）加 1，开始取指。

指令的执行过程如下：在 Q1 周期，将所取指令锁存到指令寄存器 （IR）。在随后的 Q2、Q3 和 Q4 周期中译码并执行该指令。其中读数据存储器 （读操作数）发生在 Q2 周期，写操作发生在 Q4 周期 （写目标）。

图 5-3： 时钟 / 指令周期



例 5-3： 指令流水线流程



	TCY0	TCY1	TCY2	TCY3	TCY4	TCY5
1. MOVLW 55h	取指 1	执行 1				
2. MOVWF PORTB		取指 2	执行 2			
3. BRA SUB_1			取指 3	执行 3		
4. BSF PORTA, BIT3 (Forced NOP)				取指 4	清除 （NOP）	
5. Instruction @ address SUB_1					取指 SUB_1	执行 SUB_1

除程序转移指令外，所有的指令都是单周期指令。由于程序转移指令将导致一条已取指令从流水线清除，需要重新取指，然后执行指令，所以程序转移指令需要两个周期。

5.2.3 程序存储器中的指令

程序存储器按字节寻址。指令以 2 字节或 4 字节的形式存储在程序存储器中。指令字的最低有效字节始终存储在地址为偶数的程序存储器单元中 （LSB = 0）。要保证正确指向指令单元， PC 必须以 2 为单位递增，并且 LSB 总是 0 （见第 **5.1.1** 节 **"程序计数器"**）。

图 5-4 给出了指令字存储在程序存储器中的一个示例。

CALL 和 GOTO 指令在指令中嵌入了程序存储器的绝对地址。由于指令总是存储为一个字长，因而指令所包含的数据为一个字地址。字地址会写入 PC<20:1>，由 PC 在程序存储器中访问目标地址。图 5-4 中的指令 #2 给出了指令 GOTO 0006h 在程序存储器中的译码过程。程序转移指令也采取同样的方式对相对地址偏移量进行译码。在转移指令中的偏移量代表单字指令数，PC 将以此作为偏移量跳转到指定的地址单元。第 **24.0** 节 **"指令集综述"**提供了指令集的更多详情。

5.2.4 双字指令

标准的 PIC18 指令集有 4 个双字指令：CALL、MOVFF、GOTO 和 LSFR。这些指令第二个字的 4 个最高有效位均为 1，而它的低 12 位是立即数数据，通常为一个数据存储器地址。

指令的 4 个最高有效位为 1111，用于指定一个特殊的 NOP 指令。指令顺序执行的正确程序为：执行完第一个字之后立即按顺序访问并使用第二个字中的数据。如果由于某些原因跳过了第一个字而自行执行指令的第二个字，那么第一个字将作为一条 NOP 指令执行。如果双字指令跟在修改PC的条件指令后，就有必要执行此操作。例 5-4 说明了它的执行过程。

> 注： 欲知扩展指令集中的双字指令信息，请参见第 **5.5** 节 **"程序存储器和扩展的指令集"**。

图 5-4： 程序存储器中的指令

			LSB = 1	LSB = 0	字地址 ↓
程序存储器 字节单元 →					000000h
					000002h
					000004h
					000006h
指令 1：	MOVLW	055h	0Fh	55h	000008h
指令 2：	GOTO	0006h	EFh	03h	00000Ah
			F0h	00h	00000Ch
指令 3：	MOVFF	123h, 456h	C1h	23h	00000Eh
			F4h	56h	000010h
					000012h
					000014h

例 5-4： 双字指令

目标代码	源代码		
情形 1：			
0110 0110 0000 0000	TSTFSZ	REG1	; is RAM location 0?
1100 0001 0010 0011	MOVFF	REG1, REG2	; No, skip this word
1111 0100 0101 0110			; Execute this word as a NOP
0010 0100 0000 0000	ADDWF	REG3	; continue code
情形 2：			
目标代码	源代码		
0110 0110 0000 0000	TSTFSZ	REG1	; is RAM location 0?
1100 0001 0010 0011	MOVFF	REG1, REG2	; Yes, execute this word
1111 0100 0101 0110			; 2nd word of instruction
0010 0100 0000 0000	ADDWF	REG3	; continue code

5.3 数据存储器构成

注：	当使能了 PIC18 扩展指令集时，数据存储器某些方面的操作会有所改变。欲知更多信息，请参见**第 5.6 节 “数据存储器和扩展的指令集”**。

PIC18 器件中的数据存储器是用静态 RAM 实现的。在数据存储器中，每个寄存器有 12 位地址，可存储数据达 4096 个字节。存储器空间被分为 16 个存储区，每个存储区包含 256 个字节。但 PIC18F6390/6490/8390/8490 器件只用到4个存储区。图 5-5 显示了 PIC18F6390/6490/8390/8490 器件的数据存储器构成。

数据存储器由特殊功能寄存器（Special Function Register，SFR）和通用寄存器（General Purpose Register，GPR）组成。SFR 用于单片机和外设功能模块的控制和状态显示，GPR则用于在用户应用程序中存储数据和临时存储操作的中间结果。任何未用单元的读取值均为 0。

这样的指令集和架构支持跨存储区的操作。可以通过直接、间接或变址寻址模式访问整个数据存储器。本章后面的部分将讨论寻址模式。

为了确保能在一个周期存取常用寄存器（SFR 和所选的 GPR），PIC18 器件设置了一个快速操作存取区。该存取区是一个 256 字节的存储器空间，它可实现对 SFR 和 GPR 存储区 0 的低地址单元的快速存取，而无需使用 BSR。**第 5.3.2 节“快速操作存储区”**提供了对于快速操作 RAM 的详细说明。

5.3.1 存储区选择寄存器

存储容量较大的数据存储器需要有效的寻址机制，以便对所有地址进行快速存取。理想状况下，这意味着不必为每次读写操作提供完整地址。PIC18 器件是使用 RAM 区存储机制实现快速存取的。这种机制将存储器空间分成连续的 16 个 256 字节的存储区。根据不同的指令，可以通过完整的 12 位地址直接寻址每个单元，或通过8位的低字节地址和4位存储区指针间接寻址每个单元。

PIC18 指令集中的大部分指令都使用存储区指针，也就是存储区选择寄存器（Bank Select Register，BSR）。SFR 保存单元地址的高 4 位，而指令本身则包括单元地址的低 8 位。只使用 BSR 的低 4 位（BSR3:BSR0）而不使用高 4 位，高 4 位读为 0 且不能被写入。可以通过使用 MOVLB 指令直接装载 BSR。

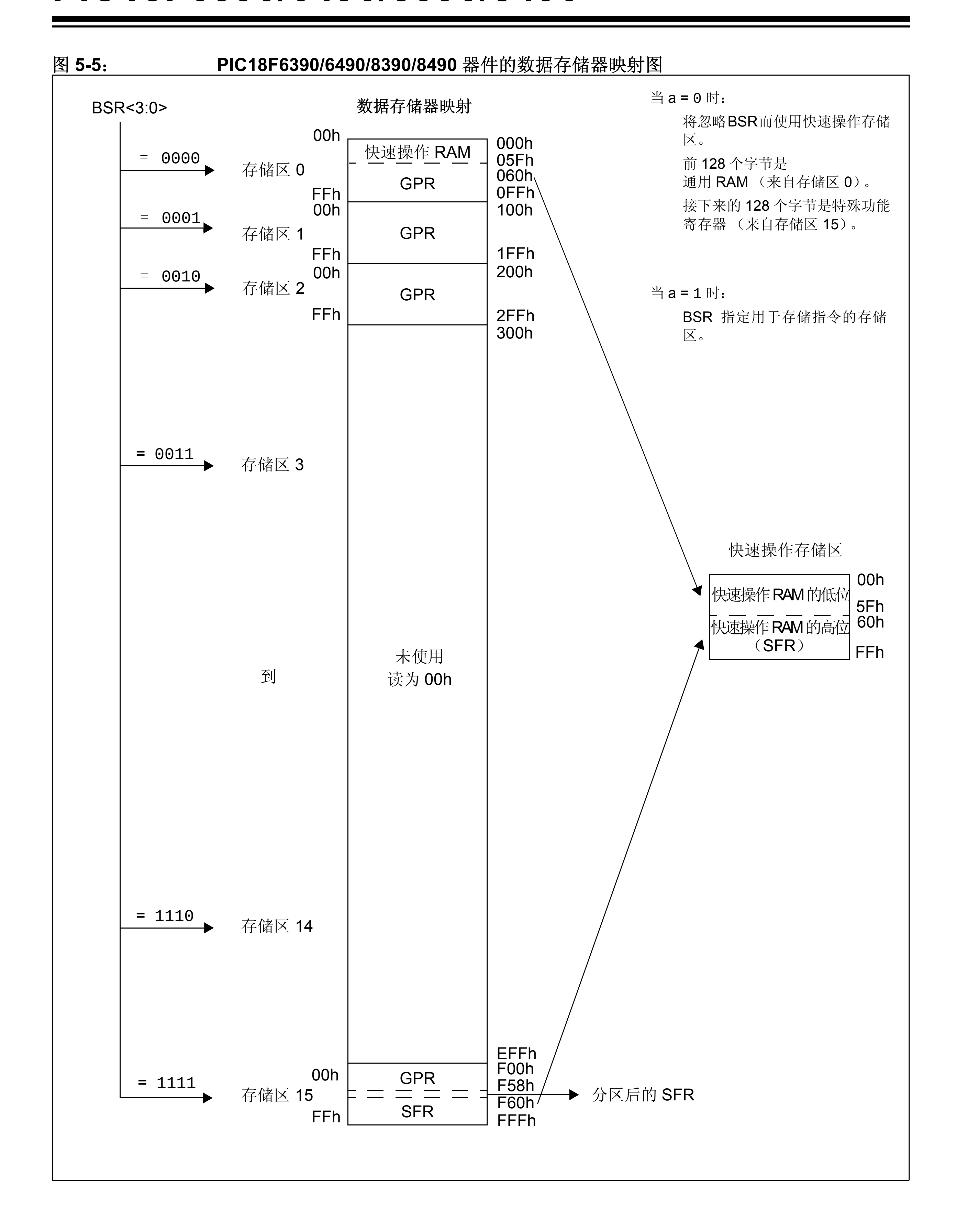
BSR 的值代表数据存储器中的存储区。指令中的 8 位指向存储区中的存储单元，可以将它看作距离存储区下边界的偏移量。图 5-6 显示了 BSR 的值与存储区之间的关系。

由于最多可有 16 个寄存器共享同一个低位地址，用户必须非常仔细以确保在执行数据读或写之前选择了正确的存储区。例如，当 BSR 为 0Fh 时将程序数据写入地址为 F9h 的 8 位地址单元将终止程序计数器的复位。

当选择存储区时，只有实际可使用的存储区才可以读写。对不可使用的存储区进行的写入操作将被忽略，而读不可使用的存储区会返回 0。虽然是这样，状态寄存器仍然会受到影响。图 5-5中的数据存储器映射图指出了可使用的存储区。

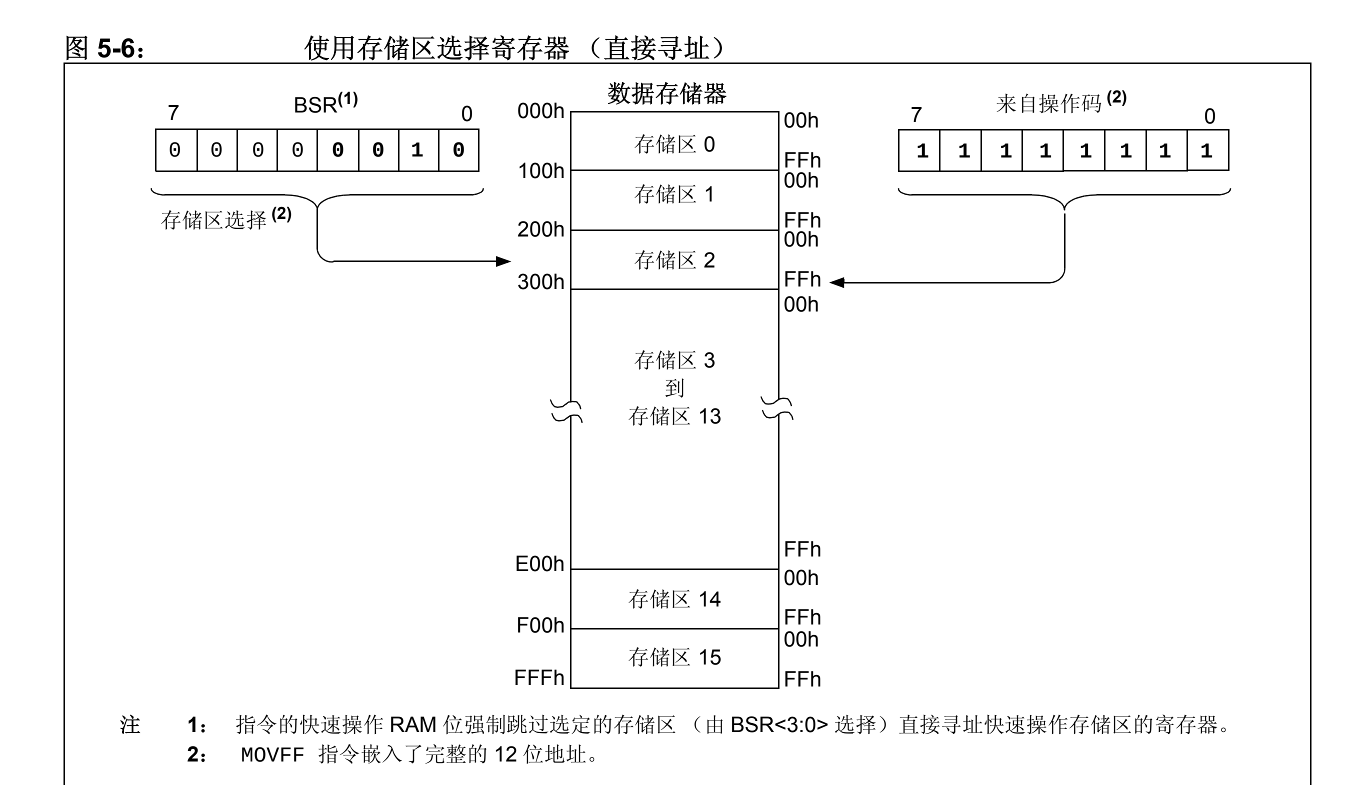
在 PIC18 的内核指令集中，只有 MOVFF 指令指定源寄存器和目标寄存器的完整 12 位地址。此指令在执行时完全忽略 BSR。所有其他指令仅包含作为操作数的低位地址，而且必须使用 BSR 或快速操作存储区来寻址目标寄存器。

图 5-5： PIC18F6390/6490/8390/8490 器件的数据存储器映射图



 初稿

图 5-6： 使用存储区选择寄存器 （直接寻址）



注 1： 指令的快速操作 RAM 位强制跳过选定的存储区 （由 BSR<3:0> 选择）直接寻址快速操作存储区的寄存器。
2： MOVFF 指令嵌入了完整的 12 位地址。

5.3.2 快速操作存储区

使用嵌入了 8 位地址的 BSR 使用户可以寻址数据存储器的整个空间，但这同时也意味着用户必须始终确保选择了正确的存储区。否则，可能会从错误的单元读取数据或将数据写入错误的单元。如果本来是向 GPR 进行写操作却将结果写入到了 SFR，后果是非常严重的。但是在每次向数据存储器进行读或写操作时确认和 / 或更改 BSR 会降低代码的执行效率。

为了连续访问大多数常用数据存储器单元，必须为数据存储器配置快速操作存储区，这样可以允许用户访问被映射的存储区而不需要指定 BSR。快速操作存储区由存储区 0 的前 96 个字节 （00h-5Fh）和 Block 15 的后 160 个字节 （60h-FF）组成。地址较低的部分被称为 “快速操作 RAM”，由 GPR 组成。地址较高的部分则被映射为器件的SFR。这两个区域被连续地映射到快速操作存储区并且可以用一个 8 位地址进行线性寻址（图 5-5）。

通过执行包括快速操作 RAM 位（指令中的“a”参数）的 PIC18 内核指令使用快速操作存储区。当 “a” 等于 “1” 时，指令使用 BSR 和包含在操作码中的 8 位地址对数据存储器寻址。而当 “a” 为 “0” 时，强制指令使用快速操作存储区地址映射，此时忽略 BSR 的当前值。

此“强制”寻址方式可使指令在一个周期内对数据地址进行操作，而不需要首先更新 BSR。这意味着用户可以更有效对 8 位地址为 80h 或以上的 SFR 进行取值和操作。地址为 60h 以下的快速操作 RAM 非常适合于存储那些用户可能需要快速访问的数据值，如直接计算结果或常用程序变量。快速操作 RAM 也可实现更加快速和有效的现场保护和变量切换。

使能扩展的指令集 （XINST 配置位 = 1）时的快速操作存储区的映射略有不同。在第 **5.6.3 节“在立即数变址寻址模式中映射快速操作存储区”**中将对此进行更详细的讨论。

5.3.3 通用寄存器

PIC18 器件可能在 GRP 区中划分了一部分存储区。这部分存储区为数据 RAM，所有指令均可访问它。GPR 区域从存储区 0 的底部（地址 000h）开始向上直到 SFR 区的底部。上电复位不会初始化 GPR，并且其他复位也不会改变其内容。

5.3.4 特殊功能寄存器

特殊功能寄存器（Special Function Register，SFR）是CPU 和外设模块用来控制器件操作的寄存器。这类寄存器以静态 RAM 的形式实现。SFR 从数据存储器的顶部（FFFh）开始向下，它占用了存储区 15 四分之三的单元空间（从 F40h 到 FFFh）。表 5-1 和表 5-2 列出了这些寄存器。

可以将 SFR 归类为两组：与“内核”器件功能（ALU、复位和中断）相关的寄存器和与外设功能相关的寄存器。在相关的章节中将对复位和中断寄存器进行说明，本章后面的部分将对 ALU 状态寄存器进行说明。与外设操作相关的寄存器将在该外设的章节中进行说明。

SFR 通常分布在外设中，用于控制相应外设的功能。未使用的 SFR 单元是不存在的，读取值为 0。

表 5-1： PIC18F6390/6490/8390/8490 器件的特殊功能寄存器映射图

地址	名称	地址	名称	地址	名称	地址	名称
FFFh	TOSU	FDFh	INDF2(1)	FBFh	CCPR1H	F9Fh	IPR1
FFEh	TOSH	FDEh	POSTINC2(1)	FBEh	CCPR1L	F9Eh	PIR1
FFDh	TOSL	FDDh	POSTDEC2(1)	FBDh	CCP1CON	F9Dh	PIE1
FFCh	STKPTR	FDCh	PREINC2(1)	FBCh	CCPR2H	F9Ch	MEMCON(3)
FFBh	PCLATU	FDBh	PLUSW2(1)	FBBh	CCPR2L	F9Bh	OSCTUNE
FFAh	PCLATH	FDAh	FSR2H	FBAh	CCP2CON	F9Ah	TRISJ(3)
FF9h	PCL	FD9h	FSR2L	FB9h	—(2)	F99h	TRISH(3)
FF8h	TBLPTRU	FD8h	STATUS	FB8h	—(2)	F98h	TRISG
FF7h	TBLPTRH	FD7h	TMR0H	FB7h	—(2)	F97h	TRISF
FF6h	TBLPTRL	FD6h	TMR0L	FB6h	—(2)	F96h	TRISE
FF5h	TABLAT	FD5h	T0CON	FB5h	CVRCON	F95h	TRISD
FF4h	PRODH	FD4h	—(2)	FB4h	CMCON	F94h	TRISC
FF3h	PRODL	FD3h	OSCCON	FB3h	TMR3H	F93h	TRISB
FF2h	INTCON	FD2h	HLVDCON	FB2h	TMR3L	F92h	TRISA
FF1h	INTCON2	FD1h	WDTCON	FB1h	T3CON	F91h	LATJ(3)
FF0h	INTCON3	FD0h	RCON	FB0h	—(2)	F90h	LATH(3)
FEFh	INDF0(1)	FCFh	TMR1H	FAFh	SPBRG1	F8Fh	LATG
FEEh	POSTINC0(1)	FCEh	TMR1L	FAEh	RCREG1	F8Eh	LATF
FEDh	POSTDEC0(1)	FCDh	T1CON	FADh	TXREG1	F8Dh	LATE
FECh	PREINC0(1)	FCCh	TMR2	FACh	TXSTA1	F8Ch	LATD
FEBh	PLUSW0(1)	FCBh	PR2	FABh	RCSTA1	F8Bh	LATC
FEAh	FSR0H	FCAh	T2CON	FAAh	—(2)	F8Ah	LATB
FE9h	FSR0L	FC9h	SSPBUF	FA9h	—(2)	F89h	LATA
FE8h	WREG	FC8h	SSPADD	FA8h	—(2)	F88h	PORTJ(3)
FE7h	INDF1(1)	FC7h	SSPSTAT	FA7h	—(2)	F87h	PORTH(3)
FE6h	POSTINC1(1)	FC6h	SSPCON1	FA6h	—(2)	F86h	PORTG
FE5h	POSTDEC1(1)	FC5h	SSPCON2	FA5h	IPR3	F85h	PORTF
FE4h	PREINC1(1)	FC4h	ADRESH	FA4h	PIR3	F84h	PORTE
FE3h	PLUSW1(1)	FC3h	ADRESL	FA3h	PIE3	F83h	PORTD
FE2h	FSR1H	FC2h	ADCON0	FA2h	IPR2	F82h	PORTC
FE1h	FSR1L	FC1h	ADCON1	FA1h	PIR2	F81h	PORTB
FE0h	BSR	FC0h	ADCON2	FA0h	PIE2	F80h	PORTA

注 1： 这不是物理寄存器。
2： 不可用的寄存器，读取值为 0。
3： 在 64 引脚的器件上不存在此寄存器。
4： 在 64 引脚的器件上存在但并不使用此寄存器。

表 5-1： PIC18F6390/6490/8390/8490 器件的特殊功能寄存器映射图 （续）

地址	名称	地址	名称	地址	名称	地址	名称
F7Fh	SPBRGH1	F6Fh	SPBRG2	F5Fh	LCDSE5(3)	F4Fh	—(2)
F7Eh	BAUDCON1	F6Eh	RCREG2	F5Eh	LCDSE4(3)	F4Eh	—(2)
F7Dh	—(2)	F6Dh	TXREG2	F5Dh	LCDSE3	F4Dh	—(2)
F7Ch	LCDDATA23(4)	F6Ch	TXSTA2	F5Ch	LCDSE2	F4Ch	—(2)
F7Bh	LCDDATA22(4)	F6Bh	RCSTA2	F5Bh	LCDSE1	F4Bh	—(2)
F7Ah	LCDDATA21	F6Ah	LCDDATA10(4)	F5Ah	LCDSE0	F4Ah	—(2)
F79h	LCDDATA20	F69h	LCDDATA9	F59h	LCDCON	F49h	—(2)
F78h	LCDDATA19	F68h	LCDDATA8	F58h	LCDPS	F48h	—(2)
F77h	LCDDATA18	F67h	LCDDATA7	F57h	—(2)	F47h	—(2)
F76h	LCDDATA17(4)	F66h	LCDDATA6	F56h	—(2)	F46h	—(2)
F75h	LCDDATA16(4)	F65h	LCDDATA5(4)	F55h	—(2)	F45h	—(2)
F74h	LCDDATA15	F64h	LCDDATA4(4)	F54h	—(2)	F44h	—(2)
F73h	LCDDATA14	F63h	LCDDATA3	F53h	—(2)	F43h	—(2)
F72h	LCDDATA13	F62h	LCDDATA2	F52h	—(2)	F42h	—(2)
F71h	LCDDATA12	F61h	LCDDATA1	F51h	—(2)	F41h	—(2)
F70h	LCDDATA11(4)	F60h	LCDDATA0	F50h	—(2)	F40h	—(2)

注 1： 这不是物理寄存器。
2： 不可用的寄存器，读取值为 0。
3： 在 64 引脚的器件上不存在此寄存器。
4： 在 64 引脚的器件上存在但并不使用此寄存器。

表 **5-2**： **PIC18F6390/6490/8390/8490** 寄存器汇总

寄存器名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	POR 与 BOR 时的值	详情请见：（页）
TOSU	—	—	—	栈顶最高字节 （TOS<20:16>）					---0 0000	59, 66
TOSH	栈顶次高字节 （TOS<15:8>）								0000 0000	59, 66
TOSL	栈顶低字节 （TOS<7:0>）								0000 0000	59, 66
STKPTR	STKFUL	STKUNF	—	返回堆栈指针					00-0 0000	59, 67
PCLATU	—	—	—	PC<20:16> 的保持寄存器					---0 0000	59, 66
PCLATH	PC<15:8> 的保持寄存器								0000 0000	59, 66
PCL	PC 低字节 （PC<7:0>）								0000 0000	59, 66
TBLPTRU	—	—	bit 21	程序存储器表指针最高字节 （TBLPTR<20:16>）					--00 0000	59, 88
TBLPTRH	程序存储器表指针次高字节 （TBLPTR<15:8>）								0000 0000	59, 88
TBLPTRL	程序存储器表指针低字节 （TBLPTR<7:0>）								0000 0000	59, 88
TABLAT	程序存储器表锁存器								0000 0000	59, 88
PRODH	乘积寄存器高字节								xxxx xxxx	59, 91
PRODL	乘积寄存器低字节								xxxx xxxx	59, 91
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x	59, 95
INTCON2	$\overline{\text{RBPU}}$	INTEDG0	INTEDG1	INTEDG2	INTEDG3	TMR0IP	INT3IP	RBIP	1111 1111	59, 96
INTCON3	INT2IP	INT1IP	INT3IE	INT2IE	INT1IE	INT3IF	INT2IF	INT1IF	1100 0000	59, 97
INDF0	使用 FSR0 的内容寻址数据存储器－ FSR0 的值不改变 （不是物理存储器）								N/A	59, 82
POSTINC0	使用 FSR0 的内容寻址数据存储器－ FSR0 的值后增 （不是物理寄存器）								N/A	59, 83
POSTDEC0	使用 FSR0 的内容寻址数据存储器－ FSR0 的值后减 （不是物理寄存器）								N/A	59, 83
PREINC0	使用 FSR0 的内容寻址数据存储器－ FSR0 的值预增 （不是物理寄存器）								N/A	59, 83
PLUSW0	使用 FSR0 的内容寻址数据存储器－ FSR0 的值预增 （不是物理寄存器）， FSR0 的偏移量由 W 寄存器提供								N/A	59, 83
FSR0H	—	—	—	—	间接数据存储器地址指针 0 的高字节				---- xxxx	59, 82
FSR0L	间接数据存储器地址指针 0 的低字节								xxxx xxxx	59, 82
WREG	工作寄存器								xxxx xxxx	59
INDF1	使用 FSR1 的内容寻址数据存储器－ FSR1 的值不改变 （不是物理寄存器）								N/A	59, 82
POSTINC1	使用 FSR1 的内容寻址数据存储器－ FSR1 的值后增 （不是物理寄存器）								N/A	59, 83
POSTDEC1	使用 FSR1 的内容寻址数据存储器－ FSR1 的值后减 （不是物理寄存器）								N/A	59, 83
PREINC1	使用 FSR1 的内容寻址数据存储器－ FSR1 的值预增 （不是物理寄存器）								N/A	59, 83
PLUSW1	使用 FSR1 的内容寻址数据存储器－ FSR1 的值预增 （不是物理寄存器）， FSR1 的偏移量由 W 寄存器提供								N/A	59, 83
FSR1H	—	—	—	—	间接数据存储器地址指针 1 的高字节				---- xxxx	60, 82
FSR1L	间接数据存储器地址指针 1 的低字节								xxxx xxxx	60, 82
BSR	—	—	—	—	存储区选择寄存器				---- 0000	60, 71
INDF2	使用 FSR2 的内容寻址数据存储器－ FSR2 的值不改变 （不是物理存储器）								N/A	60, 82
POSTINC2	使用 FSR2 的内容寻址数据存储器－ FSR2 的值后增 （不是物理寄存器）								N/A	60, 83
POSTDEC2	使用 FSR2 的内容寻址数据存储器－ FSR2 的值后减 （不是物理寄存器）								N/A	60, 83
PREINC2	使用 FSR2 的内容寻址数据存储器－ FSR2 的值预增 （不是物理寄存器）								N/A	60, 83
PLUSW2	使用 FSR2 的内容寻址数据存储器－ FSR2 的值预增 （不是物理寄存器）， FSR2 的值由 W 寄存器提供								N/A	60, 83
FSR2H	—	—	—	—	间接数据存储器地址指针 2 的高字节				---- xxxx	60, 82
FSR2L	间接数据存储器地址指针 2 的低字节								xxxx xxxx	60, 82
STATUS	—	—	—	N	OV	Z	DC	C	---x xxxx	60, 80

图注： x = 未知， u = 不变， – = 未用， q = 取值视情况而定

注 **1**： 只有当 BOREN1:BOREN0 配置位 = 01 时， SBOREN 位才可用；否则它将被禁止并读为 0。请参见**第 4.4 节 "欠压复位 （BOR）"**。

2： 这些寄存器和 / 或位在 64 引脚的器件上不存在，读取值为 0。

3： 只有在特定振荡器配置中才可使用 PLLEN 位，否则，它将被禁止并且读取值为 0。请参见**第 2.6.4 节 "INTOSC 模式下的 PLL"**。

4： 只有在禁止主清零时 （MCLRE 配置位 = 0）才可使用 RG5 位；否则， RG5 的读取值为 0。此位是只读位。

5： 根据不同的主振荡器模式可将 RA6/RA7 及其相关的锁存及方向位分别配置为端口引脚。这些位在禁止时读为 0。

6： 在 64 引脚的器件中存在但并未使用这些寄存器。如果需要的话，可将它们用作通用数据 RAM。

表 5-2： PIC18F6390/6490/8390/8490 寄存器汇总 （续）

寄存器名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	POR 与 BOR 时的值	详情请见：（页）
TMR0H	Timer0 寄存器的高字节								0000 0000	60, 132
TMR0L	Timer0 寄存器的低字节								xxxx xxxx	60, 132
T0CON	TMR0ON	T08BIT	T0CS	T0SE	PSA	T0PS2	T0PS1	T0PS0	1111 1111	60, 131
OSCCON	IDLEN	IRCF2	IRCF1	IRCF0	OSTS	IOFS	SCS1	SCS0	0100 q000	35, 60
HLVDCON	VDIRMAG	—	IRVST	HLVDEN	HLVDL3	HLVDL2	HLVDL1	HLVDL0	0-00 0101	60, 251
WDTCON	—	—	—	—	—	—	—	SWDTEN	--- ---0	60, 288
RCON	IPEN	SBOREN(1)	—	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$	0q-1 11q0	52, 60, 107
TMR1H	Timer1 寄存器的高字节								xxxx xxxx	60, 137
TMR1L	Timer1 寄存器的低字节								xxxx xxxx	60, 137
T1CON	RD16	T1RUN	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{T1SYNC}$	TMR1CS	TMR1ON	0000 0000	60, 131
TMR2	Timer2 寄存器								0000 0000	60, 141
PR2	Timer2 周期寄存器								1111 1111	60, 141
T2CON	—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0	-000 0000	60, 141
SSPBUF	SSP 接收缓冲器 / 发送寄存器								xxxx xxxx	60, 158, 166
SSPADD	I²C™ 从动模式下的 SSP 地址寄存器。I²C 主控模式下的 SSP 波特率重载寄存器。								0000 0000	60, 166
SSPSTAT	SMP	CKE	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF	0000 0000	60, 158, 167
SSPCON1	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	0000 0000	60, 159, 168
SSPCON2	GCEN	ACKSTAT	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN	0000 0000	60, 169
ADRESH	A/D 结果寄存器的高字节								xxxx xxxx	61, 240
ADRESL	A/D 结果寄存器的低字节								xxxx xxxx	61, 240
ADCON0	—	—	CHS3	CHS2	CHS1	CHS0	GO/$\overline{DONE}$	ADON	--00 0000	61, 231
ADCON1	—	—	VCFG1	VCFG0	PCFG3	PCFG2	PCFG1	PCFG0	--00 0000	61, 232
ADCON2	ADFM	—	ACQT2	ACQT1	ACQT0	ADCS2	ADCS1	ADCS0	0-00 0000	61, 233
CCPR1H	捕捉 / 比较 /PWM 寄存器 1 的高字节								xxxx xxxx	61, 152, 155
CCPR1L	捕捉 / 比较 /PWM 寄存器 1 的低字节								xxxx xxxx	61, 152, 155
CCP1CON	—	—	DC1B1	DC1B0	CCP1M3	CCP1M2	CCP1M1	CCP1M0	--00 0000	61, 147
CCPR2H	捕捉 / 比较 /PWM 寄存器 2 的高字节								xxxx xxxx	61, 152, 155
CCPR2L	捕捉 / 比较 /PWM 寄存器 2 的低字节								xxxx xxxx	61, 152, 155
CCP2CON	—	—	DC2B1	DC2B0	CCP2M3	CCP2M2	CCP2M1	CCP2M0	--00 0000	61, 147
CVRCON	CVREN	CVROE	CVRR	CVRSS	CVR3	CVR2	CVR1	CVR0	000- 0000	61, 247
CMCON	C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0	0000 0111	61, 241
TMR3H	Timer3 寄存器的高字节								xxxx xxxx	61, 145
TMR3L	Timer3 寄存器的低字节								xxxx xxxx	61, 145
T3CON	RD16	T3CCP2	T3CKPS1	T3CKPS0	T3CCP1	$\overline{T3SYNC}$	TMR3CS	TMR3ON	0000 0000	61, 143

图注： x = 未知， u = 不变， – = 未用， q = 取值视情况而定

注 1： 只有当 BOREN1:BOREN0 配置位 = 01 时， SBOREN 位才可用；否则它将被禁止并读为 0。请参见**第 4.4 节 "欠压复位 （BOR）"**。

2： 这些寄存器和 / 或位在 64 引脚的器件上不存在，读取值为 0。

3： 只有在特定振荡器配置中才可使用 PLLEN 位，否则，它将被禁止并且读取值为 0。请参见**第 2.6.4 节 "INTOSC 模式下的 PLL"**。

4： 只有在禁止主清零时 （MCLRE 配置位 = 0）才可使用 RG5 位；否则， RG5 的读取值为 0。此位是只读位。

5： 根据不同的主振荡器模式可将 RA6/RA7 及其相关的锁存及方向位分别配置为端口引脚。这些位在禁止时读为 0。

6： 在 64 引脚的器件中存在但并未使用这些寄存器。如果需要的话，可将它们用作通用数据 RAM。

表 5-2： PIC18F6390/6490/8390/8490 寄存器汇总 （续）

寄存器名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	POR 与 BOR 时的值	详情请见：（页）
SPBRG1	EUSART1 波特率发生器								0000 0000	61, 201
RCREG1	EUSART1 接收寄存器								0000 0000	61, 208
TXREG1	EUSART1 发送寄存器								0000 0000	61, 206
TXSTA1	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	0000 0000	61, 198
RCSTA1	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x	61, 199
IPR3	—	LCDIP	RC2IP	TX2IP	—	—	—	—	-111 ----	61, 106
PIR3	—	LCDIF	RC2IF	TX2IF	—	—	—	—	-000 ----	61, 100
PIE3	—	LCDIE	RC2IE	TX2IE	—	—	—	—	-000 ----	61, 103
IPR2	OSCFIP	CMIP	—	—	BCLIP	HLVDIP	TMR3IP	CCP2IP	11-- 1111	61, 105
PIR2	OSCFIF	CMIF	—	—	BCLIF	HLVDIF	TMR3IF	CCP2IF	00-- 0000	61, 99
PIE2	OSCFIE	CMIE	—	—	BCLIE	HLVDIE	TMR3IE	CCP2IE	00-- 0000	61, 102
IPR1	—	ADIP	RC1IP	TX1IP	SSPIP	CCP1IP	TMR2IP	TMR1IP	-111 1111	61, 104
PIR1	—	ADIF	RC1IF	TX1IF	SSPIF	CCP1IF	TMR2IF	TMR1IF	-000 0000	61, 98
PIE1	—	ADIE	RC1IE	TX1IE	SSPIE	CCP1IE	TMR2IE	TMR1IE	-000 0000	61, 101
OSCTUNE	INTSRC	PLLEN(3)	—	TUN4	TUN3	TUN2	TUN1	TUN0	00-0 0000	35, 61
TRISJ(2)	PORTJ 的数据方向控制寄存器								1111 1111	62, 130
TRISH(2)	PORTH 的数据方向控制寄存器								1111 1111	62, 128
TRISG	—	—	—	PORTG 的数据方向控制寄存器					---1 1111	62, 126
TRISF	PORTF 的数据方向控制寄存器								1111 1111	62, 124
TRISE	PORTE 的数据方向控制寄存器				—	—	—	—	1111 ----	62, 121
TRISD	PORTD 的数据方向控制寄存器								1111 1111	62, 119
TRISC	PORTC 的数据方向控制寄存器								1111 1111	62, 117
TRISB	PORTB 的数据方向控制寄存器								1111 1111	62, 114
TRISA	TRISA7(5)	TRISA6(5)	PORTA 的数据方向控制寄存器						1111 1111	62, 111
LATJ(2)	读 PORTJ 数据锁存器，写 PORTJ 数据锁存器								xxxx xxxx	62, 130
LATH(2)	读 PORTH 数据锁存器，写 PORTH 数据锁存器								xxxx xxxx	62, 128
LATG	—	—	—	读 PORTG 数据锁存器，写 PORTG 数据锁存器					---x xxxx	62, 126
LATF	读 PORTF 数据锁存器，写 PORTF 数据锁存器								xxxx xxxx	62, 124
LATE	读 PORTE 数据锁存器，写 PORTE 数据锁存器				—	—	—	—	xxxx xxxx	62, 121
LATD	读 PORTD 数据锁存器，写 PORTD 数据锁存器								xxxx xxxx	62, 119
LATC	读 PORTC 数据锁存器，写 PORTC 数据锁存器								xxxx xxxx	62, 117
LATB	读 PORTB 数据锁存器，写 PORTB 数据锁存器								xxxx xxxx	62, 114
LATA	LATA7(5)	LATA6(5)	读 PORTA 数据锁存器，写 PORTA 数据锁存器						xxxx xxxx	62, 111
PORTJ(2)	读 PORTJ 引脚，写 PORTJ 数据锁存器								xxxx xxxx	62, 130
PORTH(2)	读 PORTH 引脚，写 PORTH 数据锁存器								xxxx xxxx	62, 128
PORTG	—	—	RG5(4)	读 PORTG 引脚 <4:0>，写 PORTG 数据锁存器 <4:0>					--xx xxxx	62, 126
PORTF	读 PORTF 引脚，写 PORTF 数据锁存器								xxxx xxxx	62, 124
PORTE	读 PORTE 引脚，写 PORTE 数据锁存器				—	—	—	—	xxxx xxxx	62, 121
PORTD	读 PORTD 引脚，写 PORTD 数据锁存器								xxxx xxxx	62, 119
PORTC	读 PORTC 引脚，写 PORTC 数据锁存器								xxxx xxxx	62, 117
PORTB	读 PORTB 引脚，写 PORTB 数据锁存器								xxxx xxxx	62, 114
PORTA	RA7(5)	RA6(5)	读 PORTA 引脚，写 PORTA 数据锁存器						xx0x 0000	62, 111

图注： x = 未知， u = 不变， – = 未用， q = 取值视情况而定

注 1： 只有当 BOREN1:BOREN0 配置位 = 01 时， SBOREN 位才可用；否则它将被禁止并读为 0。请参见**第 4.4 节 “欠压复位 （BOR）”**。

2： 这些寄存器和 / 或位在 64 引脚的器件上不存在，读取值为 0。

3： 只有在特定振荡器配置中才可使用 PLLEN 位，否则，它将被禁止并且读取值为 0。请参见**第 2.6.4 节 “INTOSC 模式下的 PLL”**。

4： 只有在禁止主清零时 （MCLRE 配置位 = 0）才可使用 RG5 位；否则， RG5 的读取值为 0。此位是只读位。

5： 根据不同的主振荡器模式可将 RA6/RA7 及其相关的锁存及方向位分别配置为端口引脚。这些位在禁止时读为 0。

6： 在 64 引脚的器件中存在但并未使用这些寄存器。如果需要的话，可将它们用作通用数据 RAM。

表 5-2： PIC18F6390/6490/8390/8490 寄存器汇总 （续）

寄存器名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	POR 与 BOR 时的值	详情请见：（页）
SPBRGH1	EUSART1 波特率发生器寄存器的高字节								0000 0000	62, 201
BAUDCON1	ABDOVF	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	01-0 0-00	62, 200
LCDDATA23(6)	S47C3	S46C3	S45C3	S44C3	S43C3	S42C3	S41C3	S40C3	xxxx xxxx	63, 261
LCDDATA22(6)	S39C3	S38C3	S37C3	S36C3	S35C3	S34C3	S33C3	S32C3	xxxx xxxx	63, 261
LCDDATA21	S31C3	S30C3	S29C3	S28C3	S27C3	S26C3	S25C3	S24C3	xxxx xxxx	63, 261
LCDDATA20	S23C3	S22C3	S21C3	S20C3	S19C3	S18C3	S17C3	S16C3	xxxx xxxx	63, 261
LCDDATA19	S15C3	S14C3	S13C3	S12C3	S11C3	S10C3	S09C3	S08C3	xxxx xxxx	63, 261
LCDDATA18	S07C3	S06C3	S05C3	S04C3	S03C3	S02C3	S01C3	S00C3	xxxx xxxx	63, 261
LCDDATA17(6)	S47C2	S46C2	S45C2	S44C2	S43C2	S42C2	S41C2	S40C2	xxxx xxxx	63, 261
LCDDATA16(6)	S39C2	S38C2	S37C2	S36C2	S35C2	S34C2	S33C2	S32C2	xxxx xxxx	63, 261
LCDDATA15	S31C2	S30C2	S29C2	S28C2	S27C2	S26C2	S25C2	S24C2	xxxx xxxx	63, 261
LCDDATA14	S23C2	S22C2	S21C2	S20C2	S19C2	S18C2	S17C2	S16C2	xxxx xxxx	63, 261
LCDDATA13	S15C2	S14C2	S13C2	S12C2	S11C2	S10C2	S09C2	S08C2	xxxx xxxx	63, 261
LCDDATA12	S07C2	S06C2	S05C2	S04C2	S03C2	S02C2	S01C2	S00C2	xxxx xxxx	63, 261
LCDDATA11(6)	S47C1	S46C1	S45C1	S44C1	S43C1	S42C1	S41C1	S40C1	xxxx xxxx	63, 261
SPBRG2	AUSART2 波特率发生器								0000 0000	63, 220
RCREG2	AUSART2 接收寄存器								0000 0000	63, 224
TXREG2	AUSART2 发送寄存器								0000 0000	63, 222
TXSTA2	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	0000 0010	63, 218
RCSTA2	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x	63, 219
LCDDATA10(6)	S39C1	S38C1	S37C1	S36C1	S35C1	S34C1	S33C1	S32C1	xxxx xxxx	63, 261
LCDDATA9	S31C1	S30C1	S29C1	S28C1	S27C1	S26C1	S25C1	S24C1	xxxx xxxx	63, 261
LCDDATA8	S23C1	S22C1	S21C1	S20C1	S19C1	S18C1	S17C1	S16C1	xxxx xxxx	63, 261
LCDDATA7	S15C1	S14C1	S13C1	S12C1	S11C1	S10C1	S09C1	S08C1	xxxx xxxx	63, 261
LCDDATA6	S07C1	S06C1	S05C1	S04C1	S03C1	S02C1	S01C1	S00C1	xxxx xxxx	63, 261
LCDDATA5(6)	S47C0	S46C0	S45C0	S44C0	S43C0	S42C0	S41C0	S40C0	xxxx xxxx	63, 261
LCDDATA4(6)	S39C0	S38C0	S37C0	S36C0	S35C0	S34C0	S33C0	S32C0	xxxx xxxx	63, 261
LCDDATA3	S31C0	S30C0	S29C0	S28C0	S27C0	S26C0	S25C0	S24C0	xxxx xxxx	63, 261
LCDDATA2	S23C0	S22C0	S21C0	S20C0	S19C0	S18C0	S17C0	S16C0	xxxx xxxx	63, 261
LCDDATA1	S15C0	S14C0	S13C0	S12C0	S11C0	S10C0	S09C0	S08C0	xxxx xxxx	63, 261
LCDDATA0	S07C0	S06C0	S05C0	S04C0	S03C0	S02C0	S01C0	S00C0	xxxx xxxx	63, 261
LCDSE5(2)	SE47	SE46	SE45	SE44	SE43	SE42	SE41	SE40	0000 0000	64, 261
LCDSE4(2)	SE39	SE38	SE37	SE36	SE35	SE34	SE33	SE32	0000 0000	64, 260
LCDSE3	SE31	SE30	SE29	SE28	SE27	SE26	SE25	SE24	0000 0000	64, 261
LCDSE2	SE23	SE22	SE21	SE20	SE19	SE18	SE17	SE16	0000 0000	64, 261
LCDSE1	SE15	SE14	SE13	SE12	SE11	SE10	SE9	SE8	0000 0000	64, 261
LCDSE0	SE7	SE6	SE5	SE4	SE3	SE2	SE1	SE0	0000 0000	64, 261
LCDCON	LCDEN	SLPEN	WERR	—	CS1	CS0	LMUX1	LMUX0	000- 0000	64, 258
LCDPS	WFT	BIASMD	LCDA	WA	LP3	LP2	LP1	LP0	0000 0000	64, 259

图注： x = 未知， u = 不变， – = 未用， q = 取值视情况而定

注 1： 只有当 BOREN1:BOREN0 配置位 = 01 时， SBOREN 位才可用；否则它将被禁止并读为 0。请参见**第 4.4 节 "欠压复位 （BOR）"**。

2： 这些寄存器和 / 或位在 64 引脚的器件上不存在，读取值为 0。

3： 只有在特定振荡器配置中才可使用 PLLEN 位，否则，它将被禁止并且读取值为 0。请参见**第 2.6.4 节 "INTOSC 模式下的 PLL"**。

4： 只有在禁止主清零时 （MCLRE 配置位 = 0）才可使用 RG5 位；否则， RG5 的读取值为 0。此位是只读位。

5： 根据不同的主振荡器模式可将 RA6/RA7 及其相关的锁存及方向位分别配置为端口引脚。这些位在禁止时读为 0。

6： 在 64 引脚的器件中存在但并未使用这些寄存器。如果需要的话，可将它们用作通用数据 RAM。

5.3.5 STATUS 寄存器

如寄存器 5-2 所示， Status 寄存器包含 ALU 的算术运算状态。和任何其他 SFR 一样，它也可以作为任何指令的操作数。

如果一条影响 Z、DC、C、OV 或 N 位的指令以 Status 寄存器作为目标寄存器，将不会直接写入结果，而是根据指令的执行更新寄存器状态位。因此，当执行一条把 Status 寄存器作为目标寄存器的指令后，Status 寄存器的结果可能和预想的不一样。例如， CLRF STATUS 将使 Z 位置 1 并保持其余状态位不变 （000u u1uu）。

因此，建议仅使用 BCF、BSF、SWAPF、MOVFF 和 MOVWF 指令来改变 Status 寄存器，因为这些指令不会影响该寄存器中的 Z、 C、 DC、 OV 或 N 位。

欲知其他不会影响状态位的指令，请参见表 24-2 和表 24-3 中的指令集综述。

> 注： 在减法运算中，C 和 DC 位分别作为$\overline{\text{借位}}$和辅助借位标志位。

寄存器 5-2： STATUS 寄存器

U-0	U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	—	—	N	OV	Z	DC	C
bit 7							bit 0

bit 7-5 **未用位：**读为 0

bit 4 **N：**负数位
此位用于有符号的算术运算 （2 进制补码）。它表明结果是否为负 （ALU MSB = 1）。
1 = 结果为负
0 = 结果为正

bit 3 **OV：**溢出位
此位用于有符号的算术运算 （2 进制补码）。表明溢出了 7 位二进制数的范围，溢出将导致符号位 （bit7）发生改变。
1 = 有符号算术运算中发生溢出 （本次运算）
0 = 没有发生溢出

bit 2 **Z：**零位
1 = 算术运算或逻辑运算结果为零
0 = 算术运算或逻辑运算结果不为零

bit 1 **DC：**辅助进位 / $\overline{\text{借位}}$标志位
用于 ADDWF、 ADDLW、 SUBLW 和 SUBWF 指令：
1 = 结果的第 4 个低位发生了进位
0 = 结果的第 4 个低位未发生进位

注： 对于$\overline{\text{辅助借位}}$，极性是相反的。减法是通过加上第二个操作数的 2 进制补码来执行。对于移位指令 （RRF 和 RLF），此位来自源寄存器的 bit 4 或 bit 3。

bit 0 **C：**进位 / $\overline{\text{借位}}$标志位
用于 ADDWF、 ADDLW、 SUBLW 和 SUBWF 指令：
1 = 结果中最高位发生了进位
0 = 结果中最高位未发生进位

注： 对于$\overline{\text{借位}}$，极性是相反的。减法是通过加上第二个操作数的 2 进制补码来实现。对于移位指令 （RRF， RLF），此位来自源寄存器的最高位或最低位。

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

5.4 数据寻址模式

> **注：** 当使能 PIC18 扩展的指令集时， PIC18 内核指令集中某些指令的执行会发生改变。具体信息，请参见**第 5.6 节“数据存储器和扩展的指令集”**。

只可以用一种方法对程序存储器寻址，即通过程序计数器，但可以用多种方法对数据存储器空间寻址。大部分指令的寻址模式都是固定的。其他指令可能使用最多三种模式，根据它们所使用的操作数和是否使能了扩展指令集而定。

这些寻址模式为：

- 固有寻址
- 立即数寻址
- 直接寻址
- 间接寻址

当使能了扩展的指令集时（XINST 配置位 = 1），还可使用其他的寻址模式，即变址立即数偏移寻址模式。**第 5.6.1 节“使用立即数偏移量进行变址寻址”**将更详细地讨论它的操作。

5.4.1 固有和立即数寻址

很多 PIC18 控制指令根本不需要任何参数，执行这些指令要么对整个器件造成影响，要么仅针对一个寄存器进行操作。此寻址模式就是固有寻址。例如指令 SLEEP、RESET 和 DAW。

其他指令的工作方式与此类似但需要操作码中有其他直接的参数。由于需要一些立即数作为参数，这种寻址模式被称为立即数寻址。例如 ADDLW 和 MOVLW，它们分别向W寄存器添加或从中移除立即数值。其他的立即数寻址指令，例如 CALL 和 GOTO，它们包括 20 位的程序存储器地址。

5.4.2 直接寻址

直接寻址在操作码中指定操作的全部或部分源地址和 / 或目标地址。此选项由指令附带的参数指定。

在 PIC18 的内核指令集中，针对位和针对字节的指令默认情况下使用直接寻址。所有这些指令都包含某个 8 位的直接地址作为它们的最低有效字节。此地址指定数据 RAM 的某个存储区中寄存器的地址（**第 5.3.3 节“通用寄存器”**）或快速操作存储区（**第 5.3.2 节“快速操作存储区”**）中作为指令数据源的单元地址。

快速操作 RAM 位“a”确定地址的解析方式。当“a”为 1 时，BSR（**第 5.3.1 节“存储区选择寄存器”**）的内容将和指令中指定的直接地址一起用于确定寄存器的完整 12 位地址。当“a”为 0 时，此直接地址将被解析为快速操作存储区中的一个寄存器。使用快速操作 RAM 的寻址模式有时也被称为直接强制寻址模式。

有几个指令，比如 MOVFF，在操作码中包含完整的 12 位地址，既有源地址也有目标地址。在这些情况下，BSR 被完全忽略。

操作目标地址由目标位“d”确定。当“d”为 1 时，结果被存储回源寄存器并覆盖原来的内容。当“d”为“0”时，结果被存储在 W 寄存器中。没有“d”参数的指令的目标地址是隐含的，它们是正在操作的目标寄存器或 W 寄存器。

5.4.3 间接寻址

间接寻址允许用户访问数据存储器中的单元而不需要在指令中给出一个固定的地址。这种寻址方式是通过使用文件选择寄存器（File Select Register， FSR）作为指向被读写的单元的指针实现的。由于 FSR 本身作为特殊功能寄存器位于 RAM 中，因此也可在程序控制下直接控制它们。这使得 FSR 对于在数据存储器中实现诸如表和数组等数据结构非常有用。

也可以使用间接指针操作数（ Indirect File Operand，INDF）进行间接寻址。这种操作允许自动递增、递减或偏移指针，从而自动控制指针的值。它通过使用循环提高代码执行效率，如例 5-5 所示的清零整个 RAM 存储区。它也允许用户在数据存储器中进行变址寻址和其他针对程序存储器的堆栈指针操作。

例 5-5： 使用间接寻址清零 RAM（存储区 1）

```
         LFSR    FSR0, 100h   ;
NEXT     CLRF    POSTINC0     ; Clear INDF
                              ; register then
                              ; inc pointer
         BTFSS   FSR0H, 1     ; All done with
                              ; Bank1?
         BRA     NEXT         ; NO, clear next
CONTINUE                      ; YES, continue
```

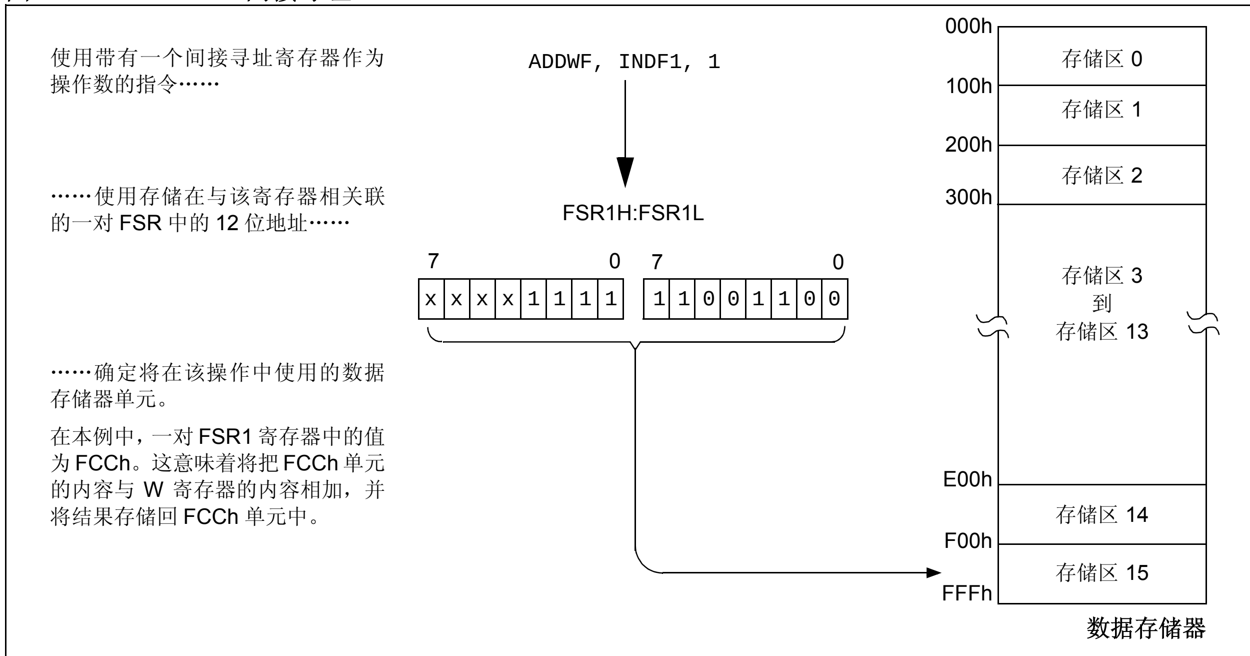
5.4.3.1 FSR 寄存器和 INDF 操作数

间接寻址的核心是三组寄存器：FSR0、FSR1和FSR2。每组寄存器都含有一对8位寄存器，FSRnH和FSRnL。FSRnH 寄存器的高 4 位未使用，因此每对 FSR 只保存一个 12 位的二进制数，从而可以线性寻址数据存储器的整个空间。因此，FSR 寄存器对被用作数据存储器的地址指针。

间接寻址是通过一组间接文件操作数（从 INDF0 到 INDF2）完成的。这些操作数可以被看作“虚拟”寄存器：它们被映射到 SFR 空间而不是通过物理方式实现的。对特定的 INDF 寄存器执行读或写操作实际上访问的是与之相应的一对 FSR 寄存器。例如，读 INDF1 就是读 FSR1H:FSR1L 指向的地址单元中的数据。使用 INDF 寄存器作为操作数的指令实际上使用的是相应的 FSR 的内容，该内容为指向目标地址的指针。INDF 操作数只是使用该指针的一种简便方法。

由于间接寻址使用完整的 12 位地址，因此没有必要进行数据 RAM 分区。所以 BSR 的当前内容和快速操作 RAM 位对于确定目标地址没有影响。

图 5-7： 间接寻址



5.4.3.2 FSR 寄存器和 POSTINC、POSTDEC、 PREINC 以及 PLUSW

除了 INDF 操作数之外，每对 FSR 寄存器还有 4 个额外的间接操作数。和 INDF 一样，它们也都是不能直接读写的“虚拟”寄存器。访问这些寄存器实际上访问的是与之相关的一对 FSR 寄存器，并在 FSR 存储的数据所指向的地址单元上进行特定的操作。这些寄存器是：

- POSTDEC：访问 FSR 的值，然后将它自动减“1”
- POSTDEC：访问 FSR 的值，然后将它自动加“1”
- PREINC：将 FSR 的值加 “1”，然后在操作中使用它。
- PLUSW：将 W 寄存器中有符号的值 （从 -127 到 128）与 FSR 寄存器中的值相加，并在操作中使用得到的新值。

在应用中使用 FSR 寄存器中的值 （不会更改此值）访问 INDF 寄存器。同样，访问 PLUSW 寄存器是将 W 寄存器中的值作为 FSR 的偏移量。该操作不会改变这两个寄存器中值，而访问其他虚拟寄存器均会更改 FSR 寄存器的值。

使用 POSTDEC、 POSTINC 和 PREINC 对 FSR 进行操作会影响整对寄存器：FSRnL 寄存器从 FFh 到 00h 溢出并向 FSRnH 寄存器进位。但这些操作的结果不会更改 Status 寄存器中的标志位（如 Z、N 和 OV 等）。

PLUSW 寄存器可以用于在数据存储器空间实现变址寻址。通过控制 W 寄存器中的值，用户可以访问相对当前指针地址有固定偏移量的地址单元。在某些应用中，该功能可以被用于在数据存储器内部实现某些强大的程序控制结构，如软件堆栈。

5.4.3.3 通过 FSR 对其他 FSR 进行操作

在某些特殊情况下，间接寻址操作以其他 FSR 或虚拟寄存器作为目标。例如，使用 FSR 指向一个虚拟寄存器会导致操作不成功。假设如下特殊情况：

FSR0H:FSR0L 保存的是 INDF1 的地址 FE7h。尝试使用 INDF0 作为操作数读取 INDF1 的值，将返回 00h。尝试使用 INDF0 作为操作数写入 INDF1，将会导致执行一条 NOP。

另一方面，使用虚拟寄存器对一对 FSR 寄存器进行写操作可能会产生与预期不同的结果。在这些情形下，会将值写入一对 FSR 寄存器，但 FSR 不会有任何递增或递减。因此，写入 INDF2 或 POSTDEC2 时会把同样的值写入 FSR2H:FSR2L。

由于 FSR 是在 SFR 空间中映射的物理寄存器，所以可以通过直接寻址对它们进行操作。用户在使用这些寄存器时应该特别小心，尤其是在代码使用间接寻址的时候。

同样，通常允许通过间接寻址对所有其他的 SFR 进行操作。用户在进行此类操作时应该特别小心，以免不小心更改设置从而影响器件操作。

5.5 程序存储器和扩展的指令集

使用扩展的指令集不会影响程序存储器的操作。

使能扩展的指令集会向现有的 PIC18 指令集添加 5 个额外的双字命令：ADDFSR、CALLW、MOVSF、MOVSS 和 SUBFSR。第 **5.2.4** 节 **"双字指令"** 说明了这些指令的执行过程。

5.6 数据存储器和扩展的指令集

使能 PIC18 扩展的指令集 （XINST 配置位 = 1）明显的更改了数据存储器及其寻址的某些方面。特别是许多 PIC18 内核指令使用快速操作存储区的方式有所不同。这是由于扩展的指令集引入了对数据存储空间的新的寻址模式。此模式也改变了使用 FSR2 及其相关的操作数进行间接寻址的方式。

同样需要了解哪些部分保持不变。数据存储器空间的大小及其线性寻址方式都不会改变。SFR 映射也保持不变。PIC18 内核指令也仍然以直接和间接寻址模式进行操作；固有和立即数指令操作照旧。FSR0 和 FSR1 的间接寻址也保持不变。

5.6.1 使用立即数偏移量进行变址寻址

使能 PIC18 扩展的指令集将更改使用 FSR2 寄存器对及其相关的指针操作数进行间接寻址的方式。在适当的条件下，使用快速操作存储区的指令（即针对位和针对字节的指令）可以利用指令中的偏移量来执行变址寻址。这种特定的寻址模式被称为使用立即数偏移量的变址寻址或立即数变址寻址模式。

在使用扩展的指令集时，这种寻址模式有如下要求：

- 强制使用快速操作存储区 （"a" = 0）；

  且

- 指针地址参数要小于或等于 5Fh。

在这些条件下，指令的地址不会被解析为地址的低字节（在直接寻址中和 BSR 一起使用），或快速操作存储区中的 8 位地址，而是被解析为由 FSR2 指定的地址指针的偏移量。将该偏移量与 FSR2 的内容相加以获取操作的目标地址。

5.6.2 受立即数变址寻址模式影响的指令

任何使用直接寻址的 PIC18 内核指令均会受到立即数变址寻址模式的潜在影响。包括所有针对字节和针对位的指令，或标准 PIC18 指令集中几乎一半的指令。只有使用固有或立即数寻址模式的指令不受影响。

此外，如果针对字节和针对位的指令使用快速操作存储区 （快速操作 RAM 位为 "1"）或包含 60h 以上的地址，它们也不受影响。符合这些条件的指令会像以前一样执行。图 5-8 显示了当使能扩展的指令集时，各种寻址模式之间的对比。

那些想要在立即数变址寻址模式中使用针对字节或针对位的指令的用户，应该注意此模式下汇编语法的改变。第 **24.2.1** 节 **"扩展指令的语法"** 中将对此进行更详细的说明。

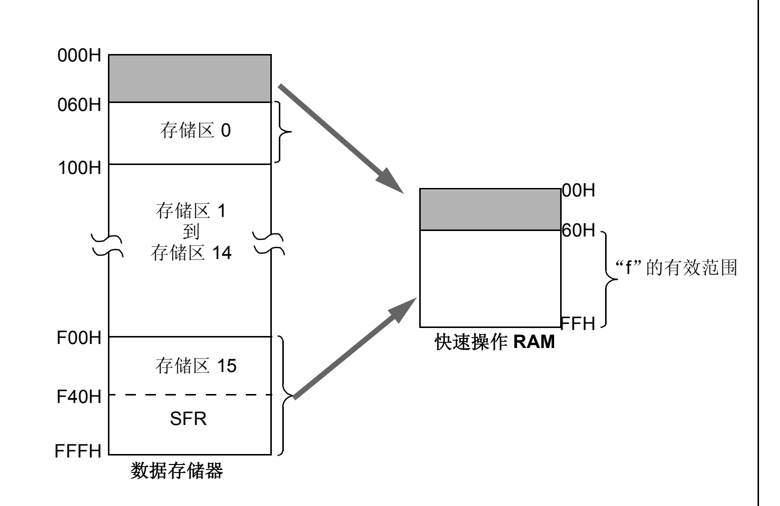
图 5-8： 针对位和针对字节的指令的寻址方式对比 （使能了扩展的指令集）

示例指令：ADDWF, f, d, a （操作码：0010 01da ffff ffff）

当 a = 0 且 f ≥ 60h 时：

此指令以直接强制模式执行。“f”被解析为 060h 到 FFFh 之间的快速操作 RAM 中的单元地址。这实际上是从 F60h 到 FFFh（存储区 15）的数据存储器单元。

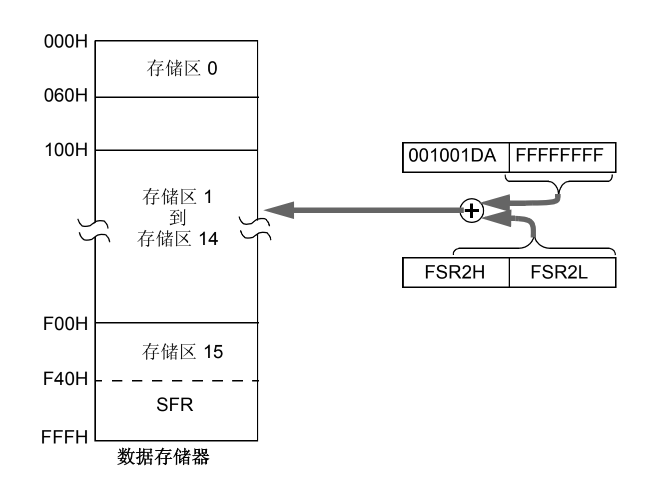
不可用此模式寻址地址低于 060h 的单元。



当 a = 0 且 f ≤ 5Fh 时：

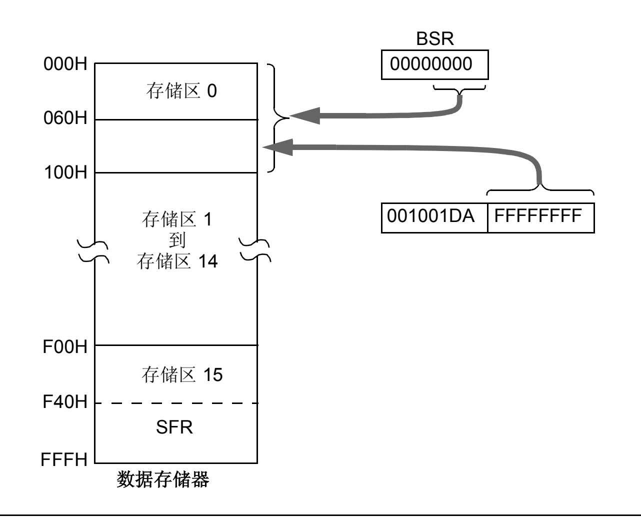
此指令以立即数变址寻址模式执行。“f”被解析为 FSR2 中地址值的偏移量。将这两个值相加以获取指令的目标寄存器的地址。此地址可以在数据存储器空间的任何地方。

注意在此模式中，正确的语法如下：
ADDWF [k], d
其中“k”就是“f”。



当 a = 1（f 可为任何值）时：

指令以直接寻址模式（也被称为直接长地址寻址模式）执行。“f”被解析为数据存储器空间的 16 个存储区中的一个单元地址。存储区由存储区选择寄存器（BSR）指定。此地址可以在数据存储器空间的任何地方。



5.6.3 在立即数变址寻址模式中映射快速操作存储区

使用立即数变址寻址模式能改变快速操作 RAM 低地址部分（00h 到 5Fh）的映射方式。此模式映射存储区 0 的内容和由用户定义的可以位于数据存储器空间中任何地方的“窗口”内容，而不是仅仅包含存储区 0 底部的内容。FSR2 的值定义映射到窗口的地址的下边界，而上边界则由 FSR2 加 95 （5Fh）决定。地址为 5Fh 以上的快速操作 RAM 的映射方法如前所述（见第 **5.3.2 节 “快速操作存储区”**）。图 5-9 显示了在此寻址模式下重新映射的快速操作存储区示例。

快速操作存储区的重新映射*仅*适用于立即数变址寻址模式。使用 BSR（快速操作 RAM 位为 “1”）的操作和前面一样继续使用直接寻址模式。任何使用间接指针操作数（包括 FR2）进行的间接或变址操作都将继续按标准的间接寻址模式进行。任何使用快速操作存储区，但寄存器地址大于 05Fh 的指令仍将使用直接寻址和常规的快速操作存储区映射。

5.6.4 立即数变址寻址模式下的 BSR

虽然当使能扩展的指令集时会重新映射快速操作存储区，但 BSR 的操作仍保持不变。操作方式与前面描述的相同，采用直接寻址模式，使用 BSR 来选择数据存储区。

图 5-9： 使用立即数变址寻址模式重新映射快速操作存储区

示例：

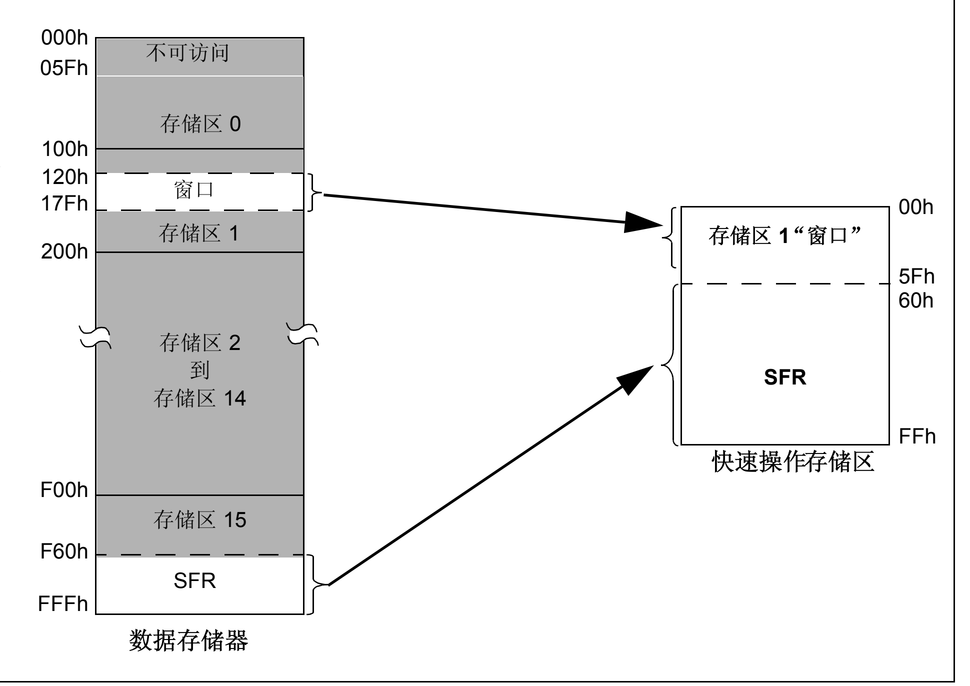
```
ADDWF f, d, a
```

FSR2H:FSR2L = 120h

从 FSR2 指针（120h）到 FSR2指针加05Fh（17Fh）区域内的存储单元被映射到快速操作 RAM 的底部（000h-05Fh）。

从 F60h 到 FFFh 的特殊功能寄存器被映射到 60h 到 FFh，和常规寻址一样。

在此模式下地址低于5Fh的存储区 0不可用。但还是可以使用 BSR 对它们进行寻址。



 初稿

6.0 闪存程序存储器

在 PIC18F6390/6490/8390/8490 器件中，程序存储器都是用只读闪存存储器实现的。正常运行期间，在整个 VDD 范围内都是可读的。对程序存储器执行读操作时，每次读取一个字节。

6.1 表读操作

PIC18 器件有两种操作方式可以在程序存储器空间和数据 RAM 之间移动字节。这两种操作分别是表读（TBLRD）和表写（TBLWT）。

表读操作从程序存储器取出数据并将数据存入 RAM 空间。图 6-1 显示了在程序存储器和数据 RAM 之间的表读操作。

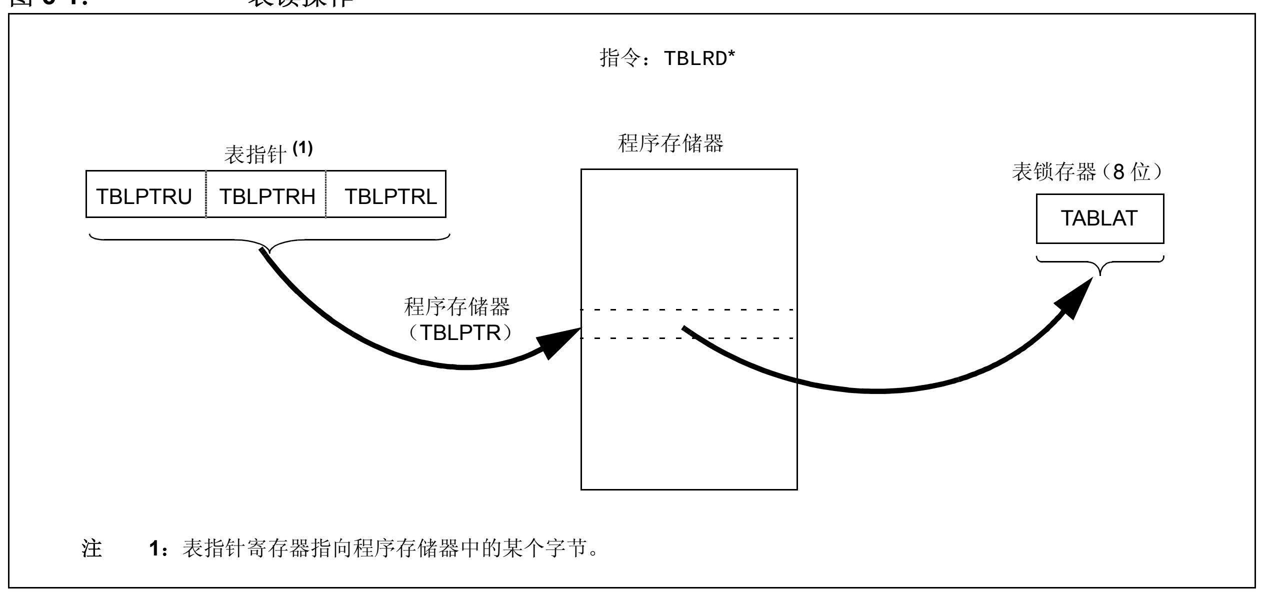
程序存储器空间是 16 位宽的，而数据 RAM 空间是 8 位宽的。表读操作和表写操作通过一个 8 位的寄存器 TABLAT 在这两者之间移动数据。

表读操作以字节为单位执行。一个包含数据而非程序指令的表块不必以字为单位组织。因此，该表块可在任何字节地址开始和结束。

由于在正常运行期间，不能写入和擦除程序存储器，在此就不讨论 TBLWT 操作了。

注	
	1：虽然TBLWT指令不能用于正常运行状态下的 PIC18F6390/6490/8390/8490 器件，但 TBLWT 指令仍然存在于指令集中。执行该指令需要两个指令周期，但实际执行的是一条 NOP 指令。
	2：只有在编程模式下 TBLWT 指令才可用并可用于在线串行编程（ICSP™）。

图 6-1： 表读操作



6.2 控制寄存器

有两个寄存器可与 TBLRD 指令结合使用：TABLAT 寄存器和 TBLPTR 寄存器组。

6.2.1 表锁存器 （TABLAT）

表锁存器（TABLAT）是一个 8 位的映射到 SFR 空间的寄存器。它用于保存程序存储器和数据 RAM 之间传输的 8 位数据。

6.2.2 表指针寄存器 （TBLPTR）

表指针寄存器 （TBLPTR）在程序存储器中寻址字节。它由 3 个 SFR 寄存器组成：表指针最高字节、表指针高字节和表指针低字节（TBLPTRU:TBLPTRH:TBLPTRL）。只使用 TBLPTRU 的最低 6 位与 TBLPTRH 和 TBLPTRL 一起形成 22 位长的指针。

TBLPTR中的值指向程序存储空间中的一个地址。低21 位允许器件寻址完整的2MB 程序存储器空间。第 22 位允许访问配置空间，包括器件 ID、用户 ID 单元和配置位。

表操作有4 种方式，当选择其中一种方式执行 TBLRD 指令时，会更新 TBLPTR 寄存器，具体采用何种方式取决于指令参数的设置。表 6-1 详细列出了这 4 种方式。这些表操作只影响 TBLPTR 的低 21 位。

当执行 TBLRD 指令时，TBLPTR 的全部 22 位用于确定哪个字节将被从程序存储器空间读到 TABLAT 中。

表 6-1： 用TBLRD指令执行表指针操作

示例	表指针操作
TBLRD*	不修改 TBLPTR
TBLRD*+	读操作后递增 TBLPTR
TBLRD*-	读操作后递减 TBLPTR
TBLRD+*	读操作前递增 TBLPTR

6.3 读闪存程序存储器

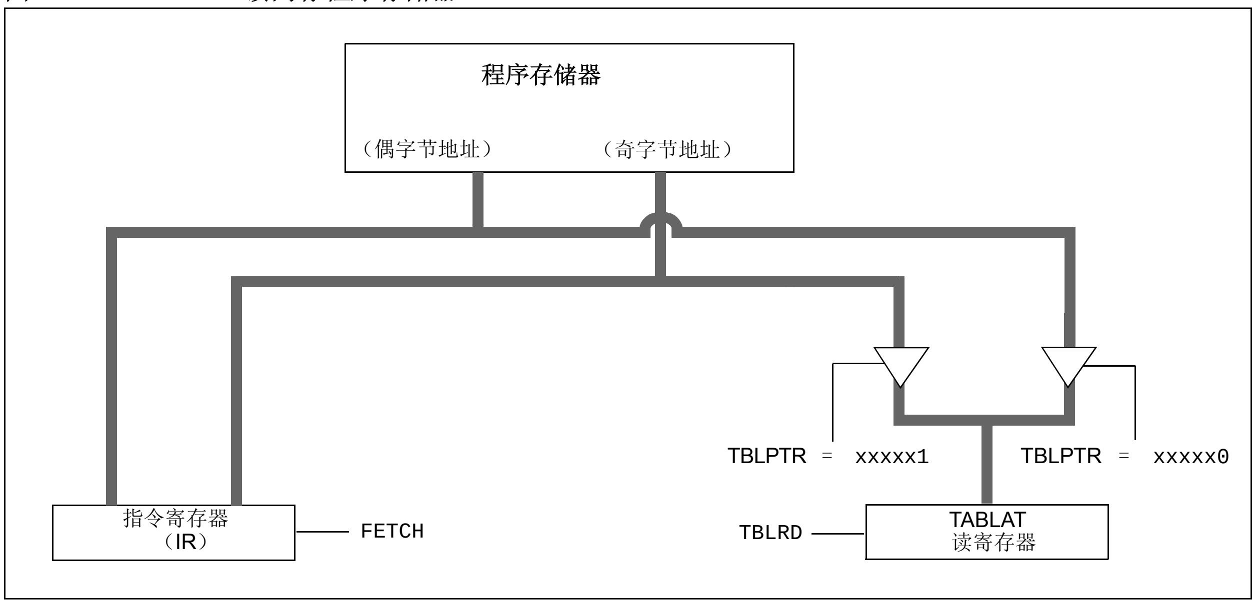
TBLRD 指令用于从程序存储器中读取数据并将数据存入数据 RAM。对程序存储器执行表读操作时每次读取一个字节。

TBLPTR 指向程序空间的某个字节。 执行 TBLRD 指令会将指向的字节存入 TABLAT。另外，可以自动修改 TBLPTR 以进行下次表读操作。

内部程序存储器通常是以字为单位组织的。由地址的最低有效位来选择字的高字节和低字节。图 6-2 显示了内部程序存储器和 TABLAT 之间的接口。

从程序存储器中读取数据的典型方法如例 6-1 所示。

图 6-2： 读闪存程序存储器



 初稿

例 **6-1**： 读取闪存程序存储器中的一个字

```
            MOVLW    CODE_ADDR_UPPER         ; Load TBLPTR with the base
            MOVWF    TBLPTRU                 ; address of the word
            MOVLW    CODE_ADDR_HIGH
            MOVWF    TBLPTRH
            MOVLW    CODE_ADDR_LOW
            MOVWF    TBLPTRL
READ_WORD
            TBLRD*+                          ; read into TABLAT and increment
            MOVF     TABLAT, W               ; get data
            MOVWF    WORD_EVEN
            TBLRD*+                          ; read into TABLAT and increment
            MOVF     TABLAT, W               ; get data
            MOVF     WORD_ODD
```

表 **6-2**： 与读程序闪存存储器操作有关的寄存器

名称	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	复位值所在页
TBLPTRU	—	—	bit 21	程序存储器表指针最高字节 （TBLPTR<20:16>）					59
TBLPTRH	程序存储器表指针次高字节 （TBLPTR<15:8>）								59
TBLPTRL	程序存储器表指针低字节 （TBLPTR<7:0>）								59
TABLAT	程序存储器表锁存器								59

图注： — = 未用位，读为 0。在访问闪存时未使用阴影单元。

注：

7.0 8 x 8 硬件乘法器

7.1 简介

所有的 PIC18 器件均包含一个 8 x 8 硬件乘法器（乘法运算器是 ALU 的一部分）。该乘法器可执行无符号运算并产生一个 16 位运算结果，该结果存储在一对乘积寄存器 PRODH:PRODL 中。该乘法器执行的运算不会影响状态寄存器中的任何标志。

通过硬件执行乘法运算只需要 1 个指令周期。硬件乘法器具有更高的计算吞吐量并减少了乘法算法的代码长度，从而可在许多先前仅能使用数字信号处理器的应用中使用PIC18器件。表 7-1给出了硬件和软件乘法运算的比较，包括所需存储器空间和执行时间。

7.2 工作原理

例 7-1 给出了一个 8 × 8 无符号乘法运算的指令序列。当已在 WREG 寄存器中装入了一个乘数时，实现该运算仅需一条指令。

例 7-2 给出了一个 8 × 8 有符号乘法运算的指令序列。要弄清乘数的符号位，必须检查每个乘数的最高有效位（MSb），并做相应的减法。

例 7-1： 8 x 8 无符号乘法程序

```
MOVF    ARG1, W     ;
MULWF   ARG2        ; ARG1 * ARG2 ->
                    ; PRODH:PRODL
```

例 7-2： 8 x 8 有符号乘法程序

```
MOVF    ARG1, W
MULWF   ARG2        ; ARG1 * ARG2 ->
                    ; PRODH:PRODL
BTFSC   ARG2, SB    ; Test Sign Bit
SUBWF   PRODH, F    ; PRODH = PRODH
                    ;         - ARG1
MOVF    ARG2, W
BTFSC   ARG1, SB    ; Test Sign Bit
SUBWF   PRODH, F    ; PRODH = PRODH
                    ;         - ARG2
```

表 7-1： 各种乘法运算的性能比较

程序	乘法实现方法	程序存储器空间（字）	周期数（最多）	时间		
				40 MHz 时	**10 MHz 时**	**4 MHz 时**
8 × 8 无符号	软件乘法	13	69	6.9 μs	27.6 μs	69 μs
	硬件乘法	1	1	100 ns	400 ns	1 μs
8 × 8 有符号	软件乘法	33	91	9.1 μs	36.4 μs	91 μs
	硬件乘法	6	6	600 ns	2.4 μs	6 μs
16 × 16 无符号	软件乘法	21	242	24.2 μs	96.8 μs	242 μs
	硬件乘法	28	28	2.8 μs	11.2 μs	28 μs
16 × 16 有符号	软件乘法	52	254	25.4 μs	102.6 μs	254 μs
	硬件乘法	35	40	4.0 μs	16.0 μs	40 μs

例 7-3 给出了一个 16 × 16 无符号乘法运算的指令序列。公式 7-1 为所使用的算法。32 位结果存储在 4 个寄存器 （RES3:RES0）中。

公式 7-1： 16 x 16 无符号乘法算法

$$
\begin{aligned}
RES3:RES0 &= ARG1H:ARG1L \bullet ARG2H:ARG2L \\
&= (ARG1H \bullet ARG2H \bullet 2^{16}) + \\
&\quad (ARG1H \bullet ARG2L \bullet 2^{8}) + \\
&\quad (ARG1L \bullet ARG2H \bullet 2^{8}) + \\
&\quad (ARG1L \bullet ARG2L)
\end{aligned}
$$

例 7-3： 16 x16 无符号乘法程序

```
    MOVF    ARG1L, W
    MULWF   ARG2L           ; ARG1L * ARG2L->
                            ; PRODH:PRODL
    MOVFF   PRODH, RES1     ;
    MOVFF   PRODL, RES0     ;
;
    MOVF    ARG1H, W
    MULWF   ARG2H           ; ARG1H * ARG2H->
                            ; PRODH:PRODL
    MOVFF   PRODH, RES3     ;
    MOVFF   PRODL, RES2     ;
;
    MOVF    ARG1L, W
    MULWF   ARG2H           ; ARG1L * ARG2H->
                            ; PRODH:PRODL
    MOVF    PRODL, W        ;
    ADDWF   RES1, F         ; Add cross
    MOVF    PRODH, W        ; products
    ADDWFC  RES2, F         ;
    CLRF    WREG            ;
    ADDWFC  RES3, F         ;
;
    MOVF    ARG1H,  W       ;
    MULWF   ARG2L           ; ARG1H * ARG2L->
                            ; PRODH:PRODL
    MOVF    PRODL, W        ;
    ADDWF   RES1, F         ; Add cross
    MOVF    PRODH, W        ; products
    ADDWFC  RES2, F         ;
    CLRF    WREG            ;
    ADDWFC  RES3,   F       ;
```

例 7-4 给出了 16 × 16 有符号乘法运算的指令序列。公式 7-2 为所使用的算法。 32 位结果存储在 4 个寄存器（RES3:RES0）中。要弄清乘数的符号位，必须检查每个乘数的最高有效位 （MSb），并做相应的减法。

公式 7-2： 16 x 16 有符号乘法算法

$$
\begin{aligned}
RES3:RES0 &= ARG1H:ARG1L \bullet ARG2H:ARG2L \\
&= (ARG1H \bullet ARG2H \bullet 2^{16}) + \\
&\quad (ARG1H \bullet ARG2L \bullet 2^{8}) + \\
&\quad (ARG1L \bullet ARG2H \bullet 2^{8}) + \\
&\quad (ARG1L \bullet ARG2L) + \\
&\quad (-1 \bullet ARG2H<7> \bullet ARG1H:ARG1L \bullet 2^{16}) + \\
&\quad (-1 \bullet ARG1H<7> \bullet ARG2H:ARG2L \bullet 2^{16})
\end{aligned}
$$

例 7-4： 16 x 16 有符号乘法程序

```
    MOVF    ARG1L, W
    MULWF   ARG2L           ; ARG1L * ARG2L ->
                            ; PRODH:PRODL
    MOVFF   PRODH, RES1     ;
    MOVFF   PRODL, RES0     ;
;
    MOVF    ARG1H, W
    MULWF   ARG2H           ; ARG1H * ARG2H ->
                            ; PRODH:PRODL
    MOVFF   PRODH, RES3     ;
    MOVFF   PRODL, RES2     ;
;
    MOVF    ARG1L, W
    MULWF   ARG2H           ; ARG1L * ARG2H ->
                            ; PRODH:PRODL
    MOVF    PRODL, W        ;
    ADDWF   RES1, F         ; Add cross
    MOVF    PRODH, W        ; products
    ADDWFC  RES2, F         ;
    CLRF    WREG            ;
    ADDWFC  RES3, F         ;
;
    MOVF    ARG1H, W        ;
    MULWF   ARG2L           ; ARG1H * ARG2L ->
                            ; PRODH:PRODL
    MOVF    PRODL, W        ;
    ADDWF   RES1, F         ; Add cross
    MOVF    PRODH, W        ; products
    ADDWFC  RES2, F         ;
    CLRF    WREG            ;
    ADDWFC  RES3, F         ;
;
    BTFSS   ARG2H, 7        ; ARG2H:ARG2L neg?
    BRA     SIGN_ARG1       ; no, check ARG1
    MOVF    ARG1L, W        ;
    SUBWF   RES2            ;
    MOVF    ARG1H, W        ;
    SUBWFB  RES3
;
SIGN_ARG1
    BTFSS   ARG1H, 7        ; ARG1H:ARG1L neg?
    BRA     CONT_CODE       ; no, done
    MOVF    ARG2L, W        ;
    SUBWF   RES2            ;
    MOVF    ARG2H, W        ;
    SUBWFB  RES3
;
CONT_CODE
   :
```


8.0 中断

PIC18F6390/6490/8390/8490 器件具有多个中断源及一个中断优先级功能，该功能可以给每个中断源分配高优先级或者低优先级。高优先级中断向量位于 0008h，低优先级中断向量位于 0018h。高优先级中断事件将中断正在处理的低优先级中断。

有 13 个寄存器用于控制中断操作。这些寄存器是：

- RCON
- INTCON
- INTCON2
- INTCON3
- PIR1、 PIR2 和 PIR3
- PIE1、 PIE2 和 PIE3
- IPR1、 IPR2 和 IPR3

建议使用 MPLAB® IDE 提供的 Microchip 头文件命名这些寄存器中的位。这使得汇编器 / 编译器能够自动识别指定寄存器内的这些位。

通常，中断源有 3 个位用于控制其操作。分别是：

- **标志位**表明发生了中断事件
- **使能位**允许程序跳转到中断向量地址处执行 （当标志位置 1 时）
- **优先级位**用于选择高优先级还是低优先级

通过将 IPEN 位 （RCON<7>）置 1，可使能中断优先级功能。当使能中断优先级时，有2个全局中断使能位。将 GIEH 位 （INTCON<7>）置 1，可使能所有优先级位已置 1（高优先级）的中断。将 GIEL 位（INTCON<6>）置 1，可使能所有优先级位已清零（低优先级）的中断。当中断标志位、使能位及相应的全局中断使能位均被置 1 时，中断将根据设置的中断优先级立即跳转到地址 0008h 或 0018h。也可以通过设置相应的使能位来禁止单个中断。

当 IPEN 位清零 （默认状态）时，便会禁止中断优先级功能，并且中断是与 PICmicro® 中档系列器件兼容的。在兼容模式下，各个中断源的中断优先级位不起作用。INTCON<6> 是 PEIE 位，用于使能 / 禁止所有的外设中断源。 INTCON<7> 是 GIE 位，用于使能 / 禁止所有中断源。在兼容模式下，所有中断均跳转到 0008h。

当响应中断时， 全局中断使能位被清零以禁止其他中断。清零后的 IPEN 位就是 GIE 位。如果使用了中断优先级，这个位就是 GIEH 位或者 GIEL 位。高优先级中断源会中断低优先级中断。在处理高优先级中断时，低优先级中断将不被响应。

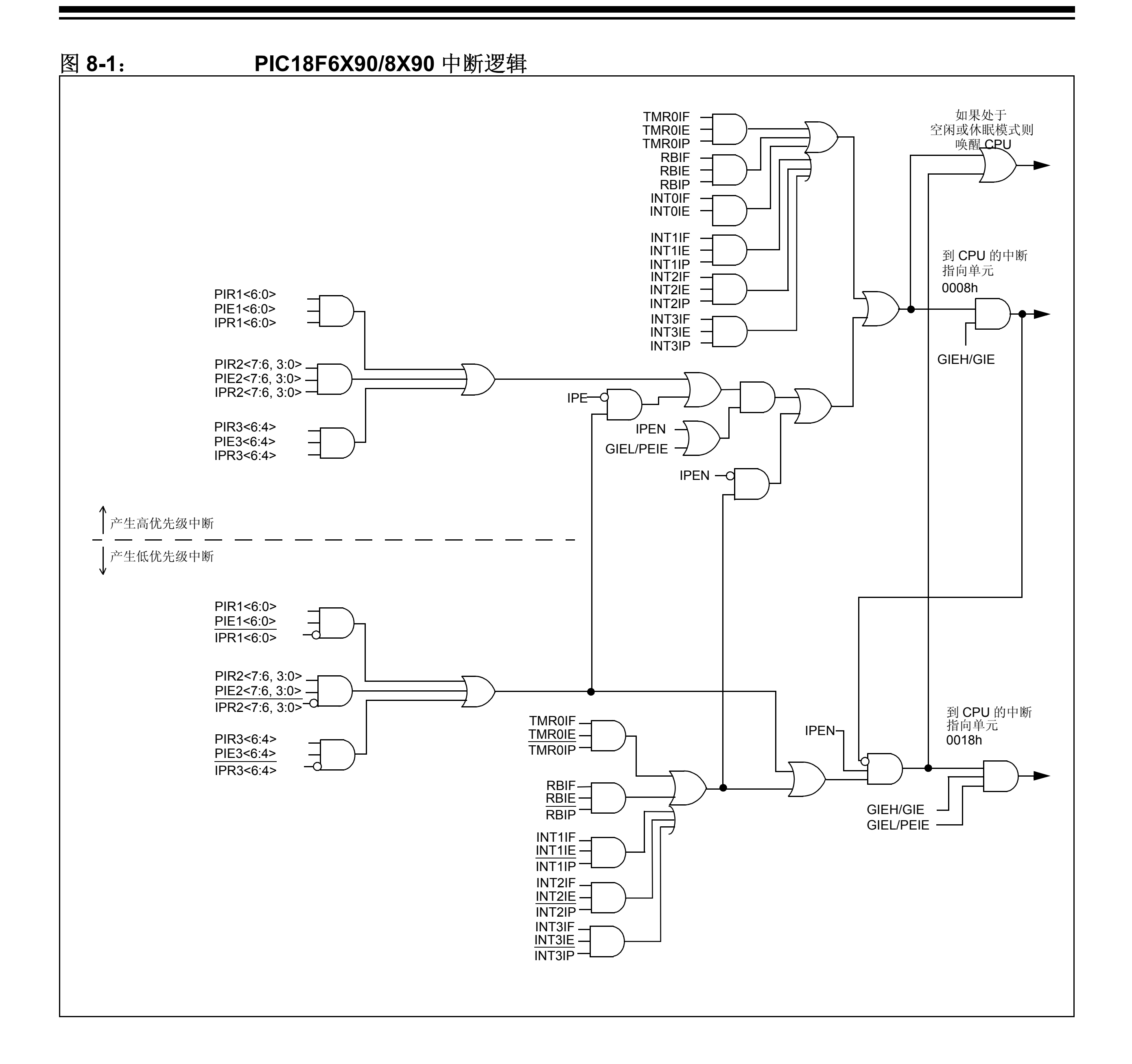
返回地址被压入堆栈，中断向量地址（0008h或0018h）被装入 PC。只要在中断服务程序中，就可以通过轮询中断标志位来确定中断源。在重新使能中断前，必须用软件将中断标志位清零，以避免重复响应这些中断。

执行“从中断返回”指令 `RETFIE` 将退出中断程序，同时将 GIE 位（若使用中断优先级则为 GIEH 或 GIEL 位）置 1，从而重新使能中断。

对于外部中断事件，例如 INT 引脚中断或者 PORTB 输入电平变化中断，中断响应延时将会是 3 到 4 个指令周期。对于单周期或双周期指令，中断响应延时完全相同。各中断标志位的置位不受对应的中断使能位和 GIE 位状态的影响。

注：	当使能**任何**中断时，不要使用 `MOVFF` 指令修改中断控制寄存器。否则可能导致单片机操作出错。

图 8-1： PIC18F6X90/8X90 中断逻辑



 初稿

8.1 INTCON 寄存器

INTCON 寄存器是可读写的寄存器，包含多个使能位、优先级位和标志位。

> 注： 当中断条件产生时，不管相应的中断使能位或全局使能位的状态如何，中断标志位都将置 1。用户软件应在使能一个中断前，先将相应的中断标志位清零。中断标志位可由软件轮询。

寄存器 8-1： INTCON: 中断控制寄存器

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-x
GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF
bit 7							bit 0

bit7 **GIE/GIEH**：全局中断使能位
当 IPEN = 0 时：
1 = 使能所有未屏蔽的中断
0 = 禁止所有中断
当 IPEN = 1 时：
1 = 使能所有的高优先级中断
0 = 禁止所有中断

bit6 **PEIE/GIEL**：外设中断使能位
当 IPEN = 0 时：
1 = 使能所有未被屏蔽的外设中断
0 = 禁止所有外设中断
当 IPEN = 1 时：
1 = 使能所有低优先级的外设中断
0 = 禁止所有低优先级的外设中断

bit5 **TMR0IE**：TMR0 溢出中断使能位
1 = 使能 TMR0 溢出中断
0 = 禁止 TMR0 溢出中断

bit4 **INT0IE**：INT0 外部中断使能位
1 = 使能 INT0 外部中断
0 = 禁止 INT0 外部中断

bit3 **RBIE**：RB 端口电平变化中断使能位
1 = 使能 RB 端口电平变化中断
0 = 禁止 RB 端口电平变化中断

bit2 **TMR0IF**：TMR0 溢出中断标志位
1 = TMR0 寄存器已经溢出 （必须用软件清零）
0 = TMR0 寄存器没有溢出

bit1 **INT0IF**：INT0 外部中断标志位
1 = 发生了 INT0 外部中断 （必须用软件清零）
0 = 未发生 INT0 外部中断

bit0 **RBIF**：RB 端口电平变化中断标志位
1 = RB7:RB4 引脚中至少有一个引脚的电平状态发生了改变 （必须用软件清零）
0 = RB7:RB4 引脚电平状态没有改变

> 注： 引脚上电平变化会不断地将 RBIF 位置 1。读取 PORTB 可以结束这种情况，并将 RBIF 位清零。

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

寄存器 8-2： INTCON2：中断控制寄存器 2

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
$\overline{\text{RBPU}}$	INTEDG0	INTEDG1	INTEDG2	INTEDG3	TMR0IP	INT3IP	RBIP
bit 7							bit 0

bit7 **$\overline{\text{RBPU}}$**：PORTB 上拉使能位
1 = 禁止所有 PORTB 上拉
0 = 根据各端口锁存器值使能 PORTB 上拉

bit6 **INTEDG0**：外部中断 0 边沿选择位
1 = 上升沿触发中断
0 = 下降沿触发中断

bit5 **INTEDG1**：外部中断 1 边沿选择位
1 = 上升沿触发中断
0 = 下降沿触发中断

bit4 **INTEDG2**：外部中断 2 边沿选择位
1 = 上升沿触发中断
0 = 下降沿触发中断

bit3 **INTEDG3**：外部中断 3 边沿选择位
1 = 上升沿触发中断
0 = 下降沿触发中断

bit2 **TMR0IP**：TMR0 溢出中断优先级位
1 = 高优先级
0 = 低优先级

bit1 **INT3IP**：INT3 外部中断优先级位
1 = 高优先级
0 = 低优先级

bit0 **RBIP**：RB 端口电平变化中断优先级位
1 = 高优先级
0 = 低优先级

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

注： 当中断条件产生时，不管相应的中断使能位或全局使能位的状态如何，中断标志位都将置 1。用户软件应在使能一个中断之前，先将相应的中断标志位清零。中断标志位可由软件轮询。

寄存器 8-3： INTCON3：中断控制寄存器 3

R/W-1	R/W-1	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
INT2IP	INT1IP	INT3IE	INT2IE	INT1IE	INT3IF	INT2IF	INT1IF
bit 7							bit 0

bit7 **INT2IP：**INT2 外部中断优先级位
1 = 高优先级
0 = 低优先级

bit6 **INT1IP：**INT1 外部中断优先级位
1 = 高优先级
0 = 低优先级

bit5 **INT3IE：**INT3 外部中断使能位
1 = 使能 INT3 外部中断
0 = 禁止 INT3 外部中断

bit4 **INT2IE：**INT2 外部中断使能位
1 = 使能 INT2 外部中断
0 = 禁止 INT2 外部中断

bit3 **INT1IE：**INT1 外部中断使能位
1 = 使能 INT1 外部中断
0 = 禁止 INT1 外部中断

bit2 **INT3IF：**INT3 外部中断标志位
1 = 发生了 INT3 外部中断 （必须用软件清零）
0 = 未发生 INT3 外部中断

bit1 **INT2IF：**INT2 外部中断标志位
1 = 发生了 INT2 外部中断 （必须用软件清零）
0 = 未发生 INT2 外部中断

bit0 **INT1IF：**INT1 外部中断标志位
1 = 发生了 INT1 外部中断 （必须用软件清零）
0 = 未发生 INT1 外部中断

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

注： 当中断条件产生时，不管相应的中断使能位或全局使能位的状态如何，中断标志位都将置 1。用户软件应在使能一个中断之前，先将相应的中断标志位清零。中断标志位可由软件轮询。

8.2 PIR 寄存器

PIR 寄存器包含各外设中断的标志位。根据外设中断源的数量，有三个外设中断请求（标志）寄存器（PIR1、PIR2 和 PIR3）。

注	
1：	当中断条件产生时，不管相应的中断使能位或全局使能位 GIE（INTCON<7>）的状态如何，中断标志位都将置 1。
2：	用户软件应在使能一个中断前和处理完一次中断后，将相应的中断标志位清零。

寄存器 8-4： PIR1：外设中断请求 （标志）寄存器 1

U-0	R/W-0	R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0
—	ADIF	RC1IF	TX1IF	SSPIF	CCP1IF	TMR2IF	TMR1IF
bit 7							bit 0

bit7 未用位：读为 0

bit6 **ADIF**：A/D 转换器中断标志位
1 = 一次 A/D 转换已完成 （必须用软件清零）
0 = A/D 转换未完成

bit5 **RC1IF**：EUSART 接收中断标志位
1 = EUSART 接收缓冲器 RC1REG 已满 （读取 RCREG 时清零）
0 = EUSART 接收缓冲器为空

bit4 **TX1IF**：EUSART 发送中断标志位
1 = EUSART 发送缓冲器 TX1REG 已空 （写入 TX1REG 时清零）
0 = EUSART 发送缓冲器为满

bit3 **SSPIF**：主同步串行口中断标志位
1 = 发送 / 接收已完成 （必须用软件清零）
0 = 等待发送 / 接收

bit2 **CCP1IF**：CCP1 中断标志位
捕捉模式：
1 = 发生了 TMR1/TMR3 寄存器捕捉（必须用软件清零）
0 = 未发生 TMR1/TMR3 寄存器捕捉
比较模式：
1 = 发生了 TMR1/TMR3 寄存器的比较匹配 （必须用软件清零）
0 = 未发生 TMR1/TMR3 寄存器的比较匹配
PWM 模式：
在此模式下未使用

bit1 **TMR2IF**：TMR2 与 PR2 匹配中断标志位
1 = TMR2 与 PR2 匹配（必须用软件清零）
0 = TMR2 与 PR2 不匹配

bit0 **TMR1IF**：TMR1 溢出中断标志位
1 = TMR1 寄存器已溢出 （必须用软件清零）
0 = TMR1 寄存器未溢出

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

寄存器 8-5： PIR2：外设中断请求 （标志）寄存器 2

R/W-0	R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
OSCFIF	CMIF	—	—	BCLIF	HLVDIF	TMR3IF	CCP2IF
bit 7							bit 0

bit7 **OSCFIF**：振荡器失效中断标志位
1 = 系统振荡器失效，改成由 INTOSC 作为时钟输入 （必须用软件清零）
0 = 系统时钟正常运行

bit6 **CMIF**：比较器中断标志位
1 = 比较器输入已改变 （必须用软件清零）
0 = 比较器输入未变化

bit5-4 未用位：读为 0

bit3 **BCLIF**：总线冲突中断标志位
1 = 发生了总线冲突 （必须用软件清零）
0 = 未发生总线冲突

bit2 **HLVDIF**：高 / 低压检测中断标志位
1 = 发生了低电压状态 （必须用软件清零）
0 = 器件电压高于低压检测跳变点

bit1 **TMR3IF**：TMR3 溢出中断标志位
1 = TMR3 寄存器已溢出 （必须用软件清零）
0 = TMR3 寄存器未溢出

bit0 **CCP2IF**：CCP2 中断标志位
捕捉模式：
1 = 发生了 TMR1/TMR3 寄存器捕捉 （必须用软件清零）
0 = 未发生 TMR1/TMR3 寄存器捕捉
比较模式：
1 = 发生了 TMR1/TMR3 寄存器的比较匹配 （必须用软件清零）
0 = 未发生 TMR1/TMR3 寄存器的比较匹配
PWM 模式：
在此模式下未使用

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

寄存器 8-6： **PIR3：外设中断请求 （标志）寄存器 3**

U-0	R/W-0	R-0	R-0	U-0	U-0	U-0	U-0
—	LCDIF	RC2IF	TX2IF	—	—	—	—
bit 7							bit 0

bit7 未用位：读为 0

bit6 **LCDIF**：LCD 中断标志位 （当选择非静态模式下的 B 波形时有效）
1 = 输出所有 COM 的 LCD 数据 （必须用软件清零）
0 = 尚未输出所有 COM 的 LCD 数据

bit5 **RC2IF**：AUSART 接收中断标志位
1 = AUSART 接收缓冲器 RC2REG 已满 （读取 RC2REG 时清零）
0 = AUSART 接收缓冲器为空

bit4 **TX2IF**：AUSART 发送中断标志位
1 = AUSART 发送缓冲器 TX2REG 已空 （写入 TX2REG 时清零）
0 = AUSART 发送缓冲器为满

bit3-0 未用位：读为 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

8.3 PIE 寄存器

PIE 寄存器包含各外设中断的使能位。根据外设中断源的数量，有三个外设中断使能寄存器（PIE1、PIE2 和 PIE3）。当 IPEN = 0 时，要使能任一外设中断，必须将 PEIE 位置 1。

寄存器 8-7： **PIE1：外设中断使能寄存器 1**

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	ADIE	RC1IE	TX1IE	SSPIE	CCP1IE	TMR2IE	TMR1IE
bit 7							bit 0

bit7 未用位：读为 0

bit6 **ADIE：**A/D 转换器中断使能位
1 = 使能 A/D 中断
0 = 禁止 A/D 中断

bit5 **RC1IE：**EUSART 接收中断使能位
1 = 使能 EUSART 接收中断
0 = 禁止 EUSART 接收中断

bit4 **TX1IE：**EUSART 发送中断使能位
1 = 使能 EUSART 发送中断
0 = 禁止 EUSART 发送中断

bit3 **SSPIE：**主同步串行口中断使能位
1 = 使能 MSSP 中断
0 = 禁止 MSSP 中断

bit2 **CCP1IE：**CCP1 中断使能位
1 = 使能 CCP1 中断
0 = 禁止 CCP1 中断

bit1 **TMR2IE：**TMR2 与 PR2 匹配中断使能位
1 = 使能 TMR2 与 PR2 匹配中断
0 = 禁止 TMR2 与 PR2 匹配中断

bit0 **TMR1IE：**TMR1 溢出中断使能位
1 = 使能 TMR1 溢出中断
0 = 禁止 TMR1 溢出中断

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

寄存器 8-8： PIE2：外设中断使能寄存器 2

R/W-0	R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
OSCFIE	CMIE	—	—	BCLIE	HLVDIE	TMR3IE	CCP2IE
bit 7							bit 0

bit7 **OSCFIE**：振荡器失效中断使能位
1 = 使能
0 = 禁止

bit6 **CMIE**：比较器中断使能位
1 = 使能
0 = 禁止

bit5-4 未用位：读为 0

bit3 **BCLIE**：总线冲突中断使能位
1 = 使能
0 = 禁止

bit2 **HLVDIE**：高 / 低压检测中断使能位
1 = 使能
0 = 禁止

bit1 **TMR3IE**：TMR3 溢出中断使能位
1 = 使能
0 = 禁止

bit0 **CCP2IE**：CCP2 中断使能位
1 = 使能
0 = 禁止

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

寄存器 8-9： PIE3：外设中断使能寄存器 3

U-0	R/W-0	R-0	R-0	U-0	U-0	U-0	U-0
—	LCDIE	RC2IE	TX2IE	—	—	—	—
bit 7							bit 0

bit7 未用位：读为 0

bit6 **LCDIE**：LCD 中断使能位 （当选择非静态模式下的 B 波形时有效）
1 = 使能
0 = 禁止

bit5 **RC2IE**：AUSART 接收中断使能位
1 = 使能
0 = 禁止

bit4 **TX2IE**：AUSART 发送中断使能位
1 = 使能
0 = 禁止

bit3-0 未用位：读为 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

8.4 IPR 寄存器

IPR 寄存器包含各外设中断的优先级位。根据外设中断源的数量，有三个外设中断优先级寄存器（IPR1、IPR2 和 IPR3）。使用优先级位时，要求将中断优先级使能（IPEN）位置 1。

寄存器 8-10： IPR1：外设中断优先级寄存器 1

U-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
—	ADIP	RC1IP	TX1IP	SSPIP	CCP1IP	TMR2IP	TMR1IP
bit 7							bit 0

bit7 未用位：读为 0

bit6 **ADIP**：A/D 转换器中断优先级位
1 = 高优先级
0 = 低优先级

bit5 **RC1IP**：EUSART 接收中断优先级位
1 = 高优先级
0 = 低优先级

bit4 **TX1IP**：EUSART 发送中断优先级位
1 = 高优先级
0 = 低优先级

bit3 **SSPIP**：主同步串行口中断优先级位
1 = 高优先级
0 = 低优先级

bit2 **CCP1IP**：CCP1 中断优先级位
1 = 高优先级
0 = 低优先级

bit1 **TMR2IP**：TMR2 与 PR2 匹配中断优先级位
1 = 高优先级
0 = 低优先级

bit0 **TMR1IP**：TMR1 溢出中断优先级位
1 = 高优先级
0 = 低优先级

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

寄存器 8-11： **IPR2：外设中断优先级寄存器 2**

R/W-1	R/W-1	U-0	U-0	R/W-1	R/W-1	R/W-1	R/W-1
OSCFIP	CMIP	—	—	BCLIP	HLVDIP	TMR3IP	CCP2IP
bit 7							bit 0

bit7 **OSCFIP：**振荡器失效中断优先级位
1 = 高优先级
0 = 低优先级

bit6 **CMIP：**比较器中断优先级位
1 = 高优先级
0 = 低优先级

bit5-4 未用位：读为 0

bit3 **BCLIP：**总线冲突中断优先级位
1 = 高优先级
0 = 低优先级

bit2 **HLVDIP：**高 / 低压检测中断优先级位
1 = 高优先级
0 = 低优先级

bit1 **TMR3IP：**TMR3 溢出中断优先级位
1 = 高优先级
0 = 低优先级

bit0 **CCP2IP：**CCP2 中断优先级位
1 = 高优先级
0 = 低优先级

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

寄存器 8-12： IPR3：外设中断优先级寄存器 3

U-0	R/W-0	R-0	R-0	U-0	U-0	U-0	U-0
—	LCDIP	RC2IP	TX2IP	—	—	—	—

bit 7 bit 0

bit7 未用位：读为 0

bit6 **LCDIP**：LCD 中断优先级位 （当选择非静态模式下的 B 波形时有效）
1 = 高优先级
0 = 低优先级

bit5 **RC2IP**：AUSART 接收中断优先级位
1 = 高优先级
0 = 低优先级

bit4 **TX2IP**：AUSART 发送中断优先级位
1 = 高优先级
0 = 低优先级

bit3-0 未用位：读为 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

8.5 RCON 寄存器

RCON寄存器中包含的位可用来确定器件上次复位或从空闲或休眠模式唤醒的原因。RCON 还包含一个可使能中断优先级的位 （IPEN）。

寄存器 8-13： RCON：复位控制寄存器

R/W-0	R/W-1	U-0	R/W-1	R-1	R-1	R/W-0	R/W-0
IPEN	SBOREN	—	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$
bit 7							bit 0

bit7 **IPEN：**中断优先级使能位
1 = 使能中断优先级
0 = 禁止中断优先级 （PIC16CXXX 兼容模式）

bit6 **SBOREN：**软件 BOR 使能位
欲知位操作和复位状态的详细信息，请参见寄存器 4-1。

bit5 **未用位：**读为 0

bit4 **$\overline{RI}$：**RESET 指令标志位
欲知位操作的详细信息，请参见寄存器 4-1。

bit3 **$\overline{TO}$：**看门狗定时器超时标志位
欲知位操作的详细信息，请参见寄存器 4-1。

bit2 **$\overline{PD}$：**掉电检测标志位
欲知位操作的详细信息，请参见寄存器 4-1。

bit1 **$\overline{POR}$：**上电复位状态位
欲知位操作的详细信息，请参见寄存器 4-1。

bit0 **$\overline{BOR}$：**欠压复位状态位
欲知位操作的详细信息，请参见寄存器 4-1。

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

8.6 INTn 引脚中断

RB0/INT0、RB1/INT1、RB2/INT2 和 RB3/INT3 引脚上的外部中断都是边沿触发的。如果 INTCON2 寄存器中相应的 INTEDGx 位被置 1，则为上升沿触发；如果该位被清零，则为下降沿触发。当 RBx/INTx 引脚上出现一个有效边沿时，相应的标志位 INTxF 被置 1。通过清零相应的使能位 INTxE，可禁止该中断。在重新使能该中断前，必须在中断服务程序中先用软件将中断标志位清零。

如果 INTxE 位在进入节能模式前被置 1，则所有的外部中断（INT0、INT1 和 INT3）均能将处理器从节能模式唤醒。如果全局中断使能位 GIE 被置 1，则处理器将在被唤醒之后转移到中断向量处执行程序。

INT1、 INT2 和 INT3 的中断优先级由中断优先级位 INT1IP（INTCON3<6>）、INT2IP（INTCON3<7>）和 INT3IP （INTCON2<1>）中的值决定。没有与 INT0 相关的优先级位。 INT0 始终是一个高优先级的中断源。

8.7 TMR0 中断

在 8 位模式（默认模式）下， TMR0 寄存器的溢出（FFh → 00h）会使 TMR0IF 标志位置 1。在 16 位模式下， TMR0H:TMR0L 寄存器对的溢出（FFFFh → 0000h）会使 TMR0IF 标志位置 1。通过将使能位 TMR0IE （INTCON<5>）置 1 或清零，可以使能或禁止该中断。 Timer0 的中断优先级由中断优先级位 TMR0IP （INTCON2<2>）中的值决定。欲进一步了解 Timer0 模块的详细内容，请参见**第 10.0 节“Timer0 模块”**。

8.8 PORTB 电平变化中断

PORTB<7:4> 上的输入电平变化会将标志位 RBIF（INTCON<0>）置 1。通过将使能位 RBIE（INTCON<3>）置 1 或清零，可以使能或禁止该中断。PORTB 电平变化中断的优先级由中断优先级位 RBIP（INTCON2<0>）中的值决定。

8.9 中断的现场保护

在中断期间，将返回的 PC 地址压入堆栈。另外，将 WREG、Status 和 BSR 寄存器的值压入快速返回堆栈。如果未使用从中断快速返回功能（见**第 5.3 节“数据存储器构成”**），那么用户可能需要在进入中断服务程序前，保存 WREG、 Status 和 BSR 寄存器的值。根据用户的具体应用，还可能需要保存其他寄存器的值。例 8-1在执行中断服务程序期间，保存并恢复WREG、Status 和 BSR 寄存器的值。

例 8-1： 将 STATUS、 WREG 和 BSR 寄存器的值保存在 RAM 中

```
MOVWF    W_TEMP                     ; W_TEMP is in virtual bank
MOVFF    STATUS, STATUS_TEMP        ; STATUS_TEMP located anywhere
MOVFF    BSR, BSR_TEMP              ; BSR_TMEP located anywhere
;
; USER ISR CODE
;
MOVFF    BSR_TEMP, BSR              ; Restore BSR
MOVF     W_TEMP, W                  ; Restore WREG
MOVFF    STATUS_TEMP, STATUS        ; Restore STATUS
```


9.0 I/O 端口

根据选定的器件和使能的功能，最多有 9 个端口可供使用。 I/O 端口的一些引脚与器件上外设功能复用。一般来说，当相应的外设被使能时，其对应的引脚就不能被用作通用 I/O 引脚。

每个端口都有三个寄存器。这些寄存器是：

- TRIS 寄存器 （数据方向寄存器）
- PORT 寄存器 （读取器件引脚的电平）
- LAT 寄存器 （输出锁存器）

在对 I/O 引脚驱动值进行读—修改—写操作时会用到数据锁存器 （LAT 寄存器）。

图 9-1 给出了通用 I/O 端口的简化模型，它没有到其他外设的接口。

图 9-1：　　通用 I/O 端口的工作原理

注　1：　I/O 引脚与 VDD 和 VSS 之间接有保护二极管。

9.1 PORTA、 TRISA 和 LATA 寄存器

PORTA 是 8 位宽的双向端口。对应的数据方向寄存器为 TRISA。将 TRISA 置 1 可以将对应的 PORTA 引脚配置为输入引脚 （即将对应的输出驱动器置于高阻态）。将 TRISA 位清 0 可以将对应的 PORTA 引脚配置为输出引脚 （即输出锁存器的数据从选定引脚输出）。读 PORTA 寄存器读入的是引脚的状态而写该寄存器是将数据写入端口锁存器。

数据锁存器（LATA）也是存储器映射的。对 LATA 寄存器执行读—修改—写操作将读写 PORTA 的输出锁存值。

RA4 引脚与 Timer0 模块的时钟输入以及 LCD 段驱动功能复用，成为 RA4/T0CKI/SEG14 引脚。 RA6 和 RA7 引脚与主振荡器引脚复用，通过在配置寄存器（如需了解详细信息，请参见**第 23.1 节“配置位”**）中对主振荡器进行配置可将这两个引脚使能为振荡器或 I/O 引脚。当没被用作端口引脚时，RA6 和 RA7 及其相关的 TRIS 和 LAT 位均读为 0。

其他的 PORTA 引脚与模拟输入、模拟 VREF+ 和 VREF- 输入引脚复用。通过将 ADCON1 寄存器 （A/D 控制寄存器 1）中的控制位清零或置 1，可将 RA3:RA0 和 RA5 引脚选作 A/D 转换器输入引脚。

通过在 CMCON 寄存器中设置相应的位还可以将 RA0 到 RA5 引脚用作比较器输入或输出。要将 RA3:RA0 用作数字输入，还必须关闭比较器。

> **注：**　在上电复位时，RA5 和 RA3:RA0 被配置为模拟输入并读为 0。而 RA4 则被配置为数字输入。

RA4/T0CKI/SEG14 引脚是施密特触发器输入和漏极开路输出，而所有其他的 PORTA 引脚都是 TTL 电平输入和 CMOS 驱动输出。

即使被用作模拟输入， TRISA 寄存器仍然控制 PORTA 引脚的方向。在将它们用作模拟输入时，用户必须确保 TRISA 寄存器中相应的位保持为置 1 状态。

RA5:RA2 还与由 LCDSE1 和 LCDSE2 寄存器中的位控制的 LCD 段驱动引脚复用。仅当禁止这些段时，才可使用 I/O 端口功能。

例 9-1：　　初始化 PORTA

```
CLRF    PORTA   ; Initialize PORTA by
                ; clearing output
                ; data latches
CLRF    LATA    ; Alternate method
                ; to clear output
                ; data latches
MOVLW   07h     ; Configure A/D
MOVWF   ADCON1  ; for digital inputs
MOVWF   07h     ; Configure comparators
MOVWF   CMCON   ; for digital input
MOVLW   0CFh    ; Value used to
                ; initialize data
                ; direction
MOVWF   TRISA   ; Set RA<3:0> as inputs
                ; RA<5:4> as outputs
```

表 9-1： PORTA 功能

引脚名称	功能	TRIS 设置	I/O	缓冲器	说明
RA0/AN0	RA0	0	O	DIG	LATA<0> 数据输出。不受模拟引脚设置的影响。
		1	I	TTL	PORTA<0> 数据输入。发生 POR 时，读为 0。
	AN0	1	I	ANA	A/D 输入通道 0。发生 POR 时为默认配置。
RA1/AN1	RA1	0	O	DIG	LATA<1> 数据输出。不受模拟引脚设置的影响。
		1	I	TTL	PORTA<1> 数据输入。发生 POR 时，读为 0。
	AN1	1	I	ANA	A/D 输入通道 1。发生 POR 时为默认配置。
RA2/AN2/VREF-/SEG16	RA2	0	O	DIG	LATA<2> 数据输出。不受模拟引脚设置的影响；当使能 LCD 段输出时被禁止。
		1	I	TTL	PORTA<2> 数据输入。发生 POR 时，读为 0。
	AN2	1	I	ANA	A/D 输入通道 2。发生 POR 时为默认配置。
	VREF-	1	I	ANA	A/D 低参考电压输入。
	SEG16	x	O	ANA	LCD 的段 16 模拟输出。
RA3/AN3/VREF+/SEG17	RA3	0	O	DIG	LATA<3> 数据输出。输出不受模拟引脚设置的影响；当使能 LCD 段输出时被禁止。
		1	I	TTL	PORTA<3> 数据输入。发生 POR 时，读为 0。
	AN3	1	I	ANA	A/D 输入通道 3。发生 POR 时为默认配置。
	VREF+	1	I	ANA	A/D 高参考电压输入。
	SEG17	x	O	ANA	LCD 的段 17 模拟输出。禁止所有其他的数字输出。
RA4/T0CKI/SEG14	RA4	0	O	DIG	LATA<4> 数据输出；当使能 LCD 段输出时被禁止。
		1	I	ST	PORTA<4> 数据输入。
	T0CKI		I	ST	Timer0 时钟输入。
	SEG14	x	O	ANA	LCD 的段 14 模拟输出。
RA5/AN4/HLVDIN/SEG15	RA5	0	O	DIG	LATA<5> 数据输出。不受模拟引脚设置的影响；当使能 LCD 段输出时被禁止。
		1	I	TTL	PORTA<5> 数据输入。发生 POR 时，读为 0。
	AN4	1	I	ANA	A/D 输入通道 5。发生 POR 时为默认配置。
	HLVDIN	1	I	ANA	高 / 低电压检测外部跳变点输入。
	SEG15	x	O	ANA	LCD 的段 15 模拟输出。
OSC2/CLKO/RA6	OSC2	x	O	ANA	主振荡器反馈输出连接 （XT、 HS 和模式）。
	CLKO	x	O	DIG	除 RCIO、INTIO2 和 ECIO 外的所有振荡器模式下的系统周期时钟输出（Fosc/4）。
	RA6	0	O	DIG	LATA<6> 数据输出。仅在 RCIO、 INTIO2 和 ECIO 模式下使能。
		1	I	TTL	PORTA<6> 数据输入。仅在 RCIO、 INTIO2 和 ECIO 模式下使能。
OSC1/CLKI/RA7	OSC1	x	I	ANA	除 INTIO 外的所有振荡器模式下的主振荡器输入连接。
	CLKI	x	I	ANA	除 INTIO 外的所有振荡器模式下的主时钟输入连接。
	RA7	0	O	DIG	LATA<7> 数据输出。仅在 INTIO 模式下可用；否则读为 0。
		1	I	TTL	PORTA<7> 数据输入。仅在 INTIO 模式下可用；否则读为 0。

图注： PWR = 供电电源， O = 输出， I = 输入， ANA = 模拟信号， DIG = 数字输出， ST = 施密特缓冲器输入，
TTL = TTL 缓冲器输入， x = 任意值 （TRIS 位不影响端口方向或在此可忽略）。

表 **9-2**： 与 **PORTA** 相关的寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
PORTA	RA7(1)	RA6(1)	RA5	RA4	RA3	RA2	RA1	RA0	62
LATA	LATA7(1)	LATA6(1)	LATA 数据输出寄存器						62
TRISA	TRISA7(1)	TRISA6(1)	PORTA 数据方向寄存器						62
ADCON1	—	—	VCFG1	VCFG0	PCFG3	PCFG2	PCFG1	PCFG0	61
LCDSE1	SE15	SE14	SE13	SE12	SE11	SE10	SE9	SE8	64
LCDSE2	SE23	SE22	SE21	SE20	SE19	SE18	SE17	SE16	64

图注： — = 未用位，读为 0。 PORTA 不使用阴影单元。

注 1： RA7:RA6 及其相关的锁存器和数据方向位根据振荡器配置使能为 I/O 引脚；否则，它们将被读为 0。

9.2 PORTB、 TRISB 和 LATB 寄存器

PORTB 是 8 位的双向端口，对应的数据方向寄存器是 TRISB。将 TRISB 位置 1 可将对应的 PORTB 引脚配置为输入引脚 （即，将对应的输出驱动器置于高阻态）。将 TRISB 位清 0 可将对应的 PORTB 引脚配置为输出引脚 （即，输出锁存器的数据从选定引脚输出）。

数据锁存器 （LATB）也是存储器映射的。对 LATB 寄存器执行读—修改—写操作将读写 PORTB 的输出锁存值。

例 9-2： 初始化 PORTB

```
CLRF    PORTB   ; Initialize PORTB by
                ; clearing output
                ; data latches
CLRF    LATB    ; Alternate method
                ; to clear output
                ; data latches
MOVLW   0CFh    ; Value used to
                ; initialize data
                ; direction
MOVWF   TRISB   ; Set RB<3:0> as inputs
                ; RB<5:4> as outputs
                ; RB<7:6> as inputs
```

PORTB 的每个引脚都有内部弱上拉电路。通过清零控制位 $\overline{\text{RBPU}}$ （INTCON2<7>）即可接通所有的上拉电路。当将 PORTB 端口的引脚配置为输出时，其弱上拉电路会自动切断。上电复位会禁止此弱上拉电路。

PORTB 的四个引脚 （RB7:RB4）具有电平变化中断功能。仅当将这些引脚配置为输入时，才可使用此中断功能（即当 RB7:RB4 中的任何一个引脚被配置为输出时，该引脚将不再具有电平变化中断功能） 。将当前 RB7:RB4 引脚上的输入电平与 PORTB 上次读入锁存器的旧值进行比较。对 RB7:RB4 引脚上的 “不匹配”输出进行或运算，产生 RB 端口电平变化中断并将标志位 RBIF （INTCON<0>）置 1。

该中断可将器件从功耗管理模式唤醒。用户可用以下方式在中断服务程序中清除该中断：

a） 读或写 PORTB （`MOVFF (ANY), PORTB` 指令除外）。这将结束不匹配状态。

b) 将标志位 RBIF 清零。

电平不匹配的状态将会一直不断地将 RBIF 标志位置 1。而读 PORTB 将结束不匹配状态，进而将 RBIF 标志位清零。

建议使用电平变化中断功能实现按键唤醒操作以及其他仅使用该中断功能的操作。在使用电平变化中断功能时，建议不要轮询 PORTB 的状态。

RB4:RB1 还与由 LCDSE1 寄存器控制的 LCD 段驱动引脚复用。仅当禁止这些段时，才可使用 I/O 端口功能。

表 9-3： PORTB 功能

引脚名称	功能	TRIS 设置	I/O	缓冲器	说明
RB0/INT0	RB0	0	O	DIG	LATB<0> 数据输出。
		1	I	TTL	PORTB<0> 数据输入；具有可编程弱上拉电路。
	INT0	1	I	ST	外部中断 0 输入。
RB1/INT1/SEG8	RB1	0	O	DIG	LATB<1> 数据输出；当使能 LCD 段输出时被禁止。
		1	I	TTL	PORTB<1> 数据输入；当 $\overline{\text{RBPU}}$ 位清零时启用弱上拉。
	INT1	1	I	ST	外部中断 1 输入。
	SEG8	x	O	ANA	LCD 的段 8 模拟输出。禁止数字输出。
RB2/INT2/SEG9	RB2	0	O	DIG	LATB<2> 数据输出；当使能 LCD 段输出时被禁止。
		1	I	TTL	PORTB<2> 数据输入；当 $\overline{\text{RBPU}}$ 位清零时启用弱上拉。
	INT2	1	I	ST	外部中断 2 输入。
	SEG9	x	O	ANA	LCD 的段 9 模拟输出
RB3/INT3/ SEG10	RB3	0	O	DIG	LATB<3> 数据输出；当使能 LCD 段输出时被禁止。
		1	I	TTL	PORTB<3> 数据输入；当 $\overline{\text{RBPU}}$ 位清零时启用弱上拉。
	INT3	1	I	ST	外部中断 3 输入。
	SEG10	x	O	ANA	LCD 的段 10 模拟输出。
RB4/KBI0/ SEG11	RB4	0	O	DIG	LATB<4> 数据输出；当使能 LCD 段输出时被禁止。
		1	I	TTL	PORTB<4> 数据输入；当 $\overline{\text{RBPU}}$ 位清零时启用弱上拉。
	KBI0	1	I	TTL	引脚电平变化中断。
	SEG11	x	O	ANA	LCD 的段 11 模拟输出。
RB5/KBI1	RB5	0	O	DIG	LATB<5> 数据输出；当使能 LCD 段输出时被禁止。
		1	I	TTL	PORTB<5> 数据输入；当 $\overline{\text{RBPU}}$ 位清零时启用弱上拉。
	KBI1	1	I	TTL	引脚电平变化中断。
RB6/KBI2/PGC	RB6	0	O	DIG	LATB<6> 数据输出；当使能 ICD 或 ICSP™ 时禁用。
		1	I	TTL	PORTB<6> 数据输入；当使能 ICD 或 ICSP 时禁用。
	KBI2	1	I	TTL	引脚电平变化中断；当使能 ICD 或 ICSP 时禁用。
	PGC	x	I	ST	供 ICSP 和 ICD 工作使用的串行执行 (ICSP) 时钟输入。[1]
RB7/KBI3/PGD	RB7	0	O	DIG	LATB<7> 数据输出；当使能 ICD 或 ICSP 时禁用。
		1	I	TTL	PORTB<7> 数据输入；当使能 ICD 或 ICSP 时禁用。
	KBI3	1	I	TTL	引脚电平变化中断；当使能 ICD 或 ICSP 时禁用。
	PGD	x	O	DIG	供 ICSP 和 ICD 工作使用的串行执行数据输出。[1]
		x	I	ST	供 ICSP 和 ICD 工作使用的串行执行数据输入。[1]

图注： PWR = 供电电源， O = 输出， I = 输入， ANA = 模拟信号， DIG = 数字输出， ST = 施密特缓冲器输入，
TTL = TTL 缓冲器输入， x = 与取值无关 （TRIS 位不影响端口方向或在此可忽略）。

注 1： 当使能 ICSP 或 ICD 时，禁止其他所有的引脚功能。

表 **9-4**： 与 **PORTB** 相关的寄存器汇总

名称	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	复位值所在页
PORTB	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	62
LATB	LATB 数据输出寄存器								62
TRISB	PORTB 数据方向寄存器								62
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
INTCON2	$\overline{\text{RBPU}}$	INTEDG0	INTEDG1	INTEDG2	INTEDG3	TMR0IP	INT3IP	RBIP	59
INTCON3	INT2IP	INT1IP	INT3IE	INT2IE	INT1IE	INT3IF	INT2IF	INT1IF	59
LCDSE1	SE15	SE14	SE13	SE12	SE11	SE10	SE9	SE8	64

图注： PORTB 不使用阴影单元。

9.3 PORTC、 TRISC 和 LATC 寄存器

PORTC 是 8 位的双向端口，对应的数据方向寄存器是 TRISC。将 TRISC 位置 1 可将对应的 PORTC 引脚配置为输入引脚 （即将对应的输出驱动为高阻态） 。将 TRISC 位清 0 可将对应的 PORTC 引脚配置为输出引脚（即输出锁存器的数据从选定引脚输出）。

数据锁存器 （LATC）也是存储器映射的。对 LATC 寄存器执行读—修改—写操作将读写 PORTC 的输出锁存值。

PORTC 与几种外设功能复用（表 9-5）。这些引脚配有施密特触发输入缓冲器。RC1 一般由配置位 CCP2MX 配置为 CCP2 模块的默认外设引脚 （默认 / 擦除的状态， CCP2MX = 1）。

当使能外设功能时，应小心定义每个 PORTC 引脚的 TRIS 位。有些外设会无视 TRIS 位的设置，将引脚定义为输出引脚或输入引脚。用户应该查阅相应的外设章节来正确设置 TRIS 位。

注：	在上电复位时，这些引脚被配置为数字输入。

外设对引脚的改写会影响 TRISC 寄存器的内容。尽管如此，读 TRISC 总是会返回其当前的内容。

RC2 和 RC5 还与由 LCDSE1 寄存器中的位控制的 LCD 段驱动引脚复用。仅当禁止这些段时，才可使用 I/O 端口功能。

例 9-3： 初始化 PORTC

```
CLRF    PORTC   ; Initialize PORTC by
                ; clearing output
                ; data latches
CLRF    LATC    ; Alternate method
                ; to clear output
                ; data latches
MOVLW   0CFh    ; Value used to
                ; initialize data
                ; direction
MOVWF   TRISC   ; Set RC<3:0> as inputs
                ; RC<5:4> as outputs
                ; RC<7:6> as inputs
```

表 9-5： PORTC 功能

引脚名称	功能	TRIS 设置	I/O	缓冲器	说明
RC0/T1OSO/T13CKI/	RC0	0	O	DIG	LATC<0> 数据输出；当使用 Timer1 振荡器时被禁止。
		1	I	ST	PORTC<0> 数据输入；当使用 Timer1 振荡器时被禁止。
	T1OSO	x	O	ANA	Timer1 振荡器输出。
	T13CKI	x	I	ST	Timer1/Timer3 时钟输入。
RC1/T1OSI/CCP2	RC1	0	O	DIG	LATC<1> 数据输出；当使用 Timer1 振荡器时被禁止。
		1	I	ST	PORTC<1> 数据输入；当使用 Timer1 振荡器时被禁止。
	T1OSI	x	I	ANA	Timer1 振荡器输入。
	CCP2(1)	0	O	DIG	CCP2 比较输出或 PWM 输出；优先于数字 I/O 数据。
		1	I	ST	CCP2 捕捉输入。
RC2/CCP1/SEG13	RC2	0	O	DIG	LATC<2> 数据输出；当使能 LCD 段时被禁止。
		1	I	ST	PORTC<2> 数据输入。
	CCP1	0	O	DIG	CCP1 比较输出或 PWM 输出；优先于数字 I/O 数据。
		1	I	ST	CCP1 捕捉输入。
	SEG13	x	O	ANA	LCD 的段 13 模拟输出。
RC3/SCK/SCL	RC3	0	O	DIG	LATC<3> 数据输出。
		1	I	ST	PORTC<3> 数据输入。
	SCK	0	O	DIG	SPI™ 时钟输出 （MSSP 模块）；优先于端口数据。
		1	I	ST	SPI 时钟输入 （MSSP 模块）。
	SCL	0	O	DIG	I^2C™ 时钟输出 （MSSP 模块）；优先于端口数据。
		1	I	ST	I^2C 时钟输入 （MSSP 模块）；输入类型取决于模块设置。
RC4/SDI/SDA	RC4	0	O	DIG	LATC<4> 数据输出。
		1	I	ST	PORTC<4> 数据输入。
	SDI	1	I	ST	SPI 数据输入 （MSSP 模块）。
	SDA	1	O	DIG	I^2C 数据输出 （MSSP 模块）；优先于端口数据。
		1	I	ST	I^2C 数据输入 （MSSP 模块）；输入类型取决于模块设置。
RC5/SDO/SEG12	RC5	0	O	DIG	LATC<5> 数据输出；当使能 LCD 段输出时被禁止。
		1	I	ST	PORTC<5> 数据输入。
	SDO	0	O	DIG	SPI 数据输出 （MSSP 模块）；优先于端口数据。
	SEG12	x	O	ANA	LCD 的段 12 模拟输出。
RC6/TX1/CK1	RC6	0	O	DIG	LATC<6> 数据输出。
		1	I	ST	PORTC<6> 数据输入。
	TX1	1	O	DIG	同步串行数据输出 （EUSART 模块）；优先于端口数据。
	CK1	1	O	DIG	同步串行数据输入 （EUSART 模块）。用户必须将其配置为输入。
		1	I	ST	同步串行时钟输入 （EUSART 模块）。
RC7/RX1/DT1	RC7	0	O	DIG	LATC<7> 数据输出。
		1	I	ST	PORTC<7> 数据输入。
	RX1	1	I	ST	异步串行接收数据输入 （EUSART 模块）。
	DT1	1	O	DIG	同步串行数据输出 （EUSART 模块）；优先于端口数据。
		1	I	ST	同步串行数据输入 （EUSART 模块）。用户必须将其配置为输入。

图注： PWR = 供电电源， O = 输出， I = 输入， ANA = 模拟信号， DIG = 数字输出， ST = 施密特缓冲器输入， TTL = TTL 缓冲器输入， x = 与取值无关 （TRIS 位不影响端口方向或在此可忽略）。

注 1： CCP2 的默认设置 （配置位 CCP2MX = 1）。

 初稿

表 **9-6：** 与 **PORTC** 相关的寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
PORTC	RC7	RC6	RC5	RC4	RC3	RC2	RC1	RC0	62
LATC	LATC 数据输出寄存器								62
TRISC	PORTC 数据方向寄存器								62
LCDSE1	SE15	SE14	SE13	SE12	SE11	SE10	SE9	SE8	64

图注： PORTC 不使用阴影单元。

9.4 PORTD、 TRISD 和 LATD 寄存器

PORTD 是 8 位的双向端口，对应的数据方向寄存器是 TRISD。将 TRISD 位置 1 可将对应的 PORTD 引脚配置为输入引脚 （即将对应的输出驱动器置于高阻态）。将 TRISD 位清 0 可将对应的 PORTD 引脚配置为输出引脚 （即输出锁存器的数据从选定引脚输出）。

数据锁存器 （LATD）也是存储器映射的。对 LATD 寄存器执行读—修改—写操作将读写 PORTD 的输出锁存值。

PORTD 上所有引脚都配有施密特触发输入缓冲器。每个引脚都可被单独地配置为输入或输出。

注：	在上电复位时，这些引脚被配置为数字输入。

PORTD 还与由 LCDSE0 寄存器控制的 LCD 段驱动引脚复用。仅当禁止这些段时，才可使用 I/O 端口功能。

例 9-4： 初始化 PORTD

```
CLRF    PORTD   ; Initialize PORTD by
                ; clearing output
                ; data latches
CLRF    LATD    ; Alternate method
                ; to clear output
                ; data latches
MOVLW   0CFh    ; Value used to
                ; initialize data
                ; direction
MOVWF   TRISD   ; Set RD<3:0> as inputs
                ; RD<5:4> as outputs
                ; RD<7:6> as inputs
```

表 9-7： PORTD 功能

引脚名称	功能	TRIS 设置	I/O	缓冲器	说明
RD0/SEG0	RD0	0	O	DIG	LATD<0> 数据输出；当使能 LCD 段时被禁止。
		1	I	ST	PORTD<0> 数据输入。
	SEG0	x	O	ANA	LCD 的段 0 模拟输出。
RD1/SEG1	RD1	0	O	DIG	LATD<1> 数据输出；当使能 LCD 段时被禁止。
		1	I	ST	PORTD<1> 数据输入。
	SEG1	x	O	ANA	LCD 的段 1 模拟输出。
RD2/SEG2	RD2	0	O	DIG	LATD<2> 数据输出；当使能 LCD 段时被禁止。
		1	I	ST	PORTD<2> 数据输入。
	SEG2	x	O	ANA	LCD 的段 2 模拟输出。
RD3/SEG3	RD3	0	O	DIG	LATD<3> 数据输出；当使能 LCD 段时被禁止。
		1	I	ST	PORTD<3> 数据输入。
	SEG3	x	O	ANA	LCD 的段 3 模拟输出。
RD4/SEG4	RD4	0	O	DIG	LATD<4> 数据输出；当使能 LCD 段时被禁止。
		1	I	ST	PORTD<4> 数据输入。
	SEG4	x	O	ANA	LCD 模块的段 4 模拟输出。
RD5/SEG5	RD5	0	O	DIG	LATD<5> 数据输出；当使能 LCD 段时被禁止。
		1	I	ST	PORTD<5> 数据输入。
	SEG5	x	O	ANA	LCD 的段 5 模拟输出。
RD6/SEG6	RD6	0	O	DIG	LATD<6> 数据输出；当使能 LCD 段时被禁止。
		1	I	ST	PORTD<6> 数据输入。
	SEG6	x	O	ANA	LCD 的段 6 模拟输出。
RD7/SEG7	RD7	0	O	DIG	LATD<7> 数据输出；当使能 LCD 段时被禁止。
		1	I	ST	PORTD<7> 数据输入。
	SEG7	x	O	ANA	LCD 的段 7 模拟输出。

图注： PWR = 供电电源， O = 输出， I = 输入， ANA = 模拟信号， DIG = 数字输出， ST = 施密特缓冲器输入， TTL = TTL 缓冲器输入， x = 与取值无关 （TRIS 位不影响端口方向或在此可忽略）。

表 **9-8**： 与 **PORTD** 相关的寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
PORTD	RD7	RD6	RD5	RD4	RD3	RD2	RD1	RD0	62
LATD	LATD 数据输出寄存器								62
TRISD	PORTD 数据方向寄存器								62
LCDSE0	SE7	SE6	SE5	SE4	SE3	SE2	SE1	SE0	64

9.5 PORTE、 TRISE 和 LATE 寄存器

PORTE 是 4 位的双向端口，对应的数据方向寄存器是 TRISE。将 TRISE 位置 1 可将对应的 PORTE 引脚配置为输入引脚 （即将对应的输出驱动器置于高阻态）。将 TRISE 位清 0 可将对应的 PORTE 引脚配置为输出引脚（即，输出锁存器的数据从选定引脚输出）。

数据锁存器 （LATE）也是存储器映射的。对 LATE 寄存器执行读—修改—写操作将读写 PORTE 的输出锁存值。

PORTE 上所有引脚都配有施密特触发输入缓冲器。每个引脚都可被单独地配置为输入或输出。

注：	在上电复位时，这些引脚被配置为数字输入。

RE6:RE4 引脚与三个 LCD 公共驱动引脚复用。具体哪些 PORTE 引脚用作 I/O 引脚取决于哪些 LCD 公共驱动端口处于激活状态。该配置由 LMUX1:LMUX0 控制位（LCDCON<1:0>）决定。表 9-9 对引脚的可用性做了总结。

RE7 与由 LCDSE3<7> 位控制的 LCD 段驱动引脚（SEG31）复用。仅当禁止该段时，才可使用 I/O 端口功能。

注：	与其他 PIC18F 器件的 RE2:RE0 对应的引脚具有 LCDBIAS3:LCDBIAS1 的功能，而与其他 PIC18F 器件的 RE3 对应的引脚具有 COM0 的功能。 这四个引脚不能被用作数字 I/O。

RE7 也可被配置为 CCP2 模块的备用外设引脚。这通过清零 CCP2MX 配置位实现。

表 9-9： 在不同的 LCD 驱动配置中可以使用的 PORTE 引脚

LCDCON <1:0>	活动的 LCD 公共端口	可用作 I/O 端口的 PORTE 引脚
00	COM0	RE6、 RE5 和 RE4
01	COM0 和 COM1	RE6 和 RE5
10	COM0、 COM1 和 COM2	RE6
11	所有 (从 COM0 到 COM3)	无

例 9-5： 初始化 PORTE

```
CLRF    PORTE   ; Initialize PORTE by
                ; clearing output
                ; data latches
CLRF    LATE    ; Alternate method
                ; to clear output
                ; data latches
MOVLW   30h     ; Value used to
                ; initialize data
                ; direction
MOVWF   TRISE   ; Set RE<5:4> as inputs
                ; RE<7:6> as outputs
```


表 9-10： PORTE 功能

引脚名称	功能	TRIS 设置	I/O	缓冲器	说明
RE4/COM1	RE4	0	O	DIG	LATE<4> 数据输出；当使能 LCD 公共端时被禁止。
		1	I	ST	PORTE<4> 数据输入。
	COM1	x	O	ANA	LCD 的公共端 1 模拟输出。
RE5/COM2	RE5	0	O	DIG	LATE<5> 数据输出；当使能 LCD 公共端时被禁止。
		1	I	ST	PORTE<5> 数据输入。
	COM2	x	O	ANA	LCD 的公共端 2 模拟输出。
RE6/COM3	RE6	0	O	DIG	LATE<6> 数据输出；当使能 LCD 段输出时被禁止。
		1	I	ST	PORTE<6> 数据输入。
	COM3	x	O	ANA	LCD 的公共端 3 模拟输出。
RE7/CCP2/SEG31	RE7	0	O	DIG	LATE<7> 数据输出；当使能 LCD 段输出时被禁止。
		1	I	ST	PORTE<7> 数据输入。
	CCP2(1)	0	O	DIG	CCP2 比较输出和 CCP2 PWM 输出；优先于端口数据。
		1	I	ST	CCP2 捕捉输入。
	SEG31	x	O	ANA	LCD 的段 31 模拟输出。

图注： PWR = 供电电源， O = 输出， I = 输入， ANA = 模拟信号， DIG = 数字输出， ST = 施密特缓冲器输入，TTL = TTL 缓冲器输入， x = 与取值无关 （TRIS 位不影响端口方向或在此可忽略）。

注 1： CCP2 的备用设置 （当配置位 CCP2MX = 0 时）。

表 9-11： 与 PORTE 相关的寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
PORTE	RE7	RE6	RE5	RE4	—	—	—	—	62
LATE	LATE 数据输出寄存器				—	—	—	—	62
TRISE	PORTE 数据方向位				—	—	—	—	62
LCDCON	LCDEN	SLPEN	WERR	—	CS1	CS0	LMUX1	LMUX0	64
LCDSE3	SE31	SE30	SE29	SE28	SE27	SE26	SE25	SE24	64

图注： — = 未用位，读为 0。 PORTE 不使用阴影单元。

9.6 PORTF、 LATF 和 TRISF 寄存器

PORTF 是 8 位的双向端口，对应的数据方向寄存器是 TRISF。将 TRISF 位置 1 可将对应的 PORTF 引脚配置为输入引脚 （即将对应的输出驱动器置于高阻态）。将 TRISF 位清 0 可将对应的 PORTF 引脚配置为输出引脚（即输出锁存器的数据从选定引脚输出）。

数据锁存器（LATF）也是存储器映射的。对 LATF 寄存器执行读—修改—写操作将读写 PORTF 的输出锁存值。

PORTF 上所有引脚都配有施密特触发输入缓冲器。每个引脚都可被单独地配置为输入或输出。

PORTF 与几个模拟外设功能复用，包括 A/D 转换器输入、比较器输入、输出以及参考电压。

注	**1：** 上电复位时，RF6:RF0 引脚被配置为输入引脚并读为 0。 **2：** 要将 PORTF 配置为数字 I/O，需关闭比较器并设置 ADCON1。

PORTF 还与由 LCDSE2 和 LCDSE3 寄存器控制的 LCD 段驱动引脚复用。仅当禁止这些段时，才可使用 I/O 端口功能。

例 9-6： 初始化 PORTF

```
CLRF    PORTF   ; Initialize PORTF by
                ; clearing output
                ; data latches
CLRF    LATF    ; Alternate method
                ; to clear output
                ; data latches
MOVLW   0x07    ;
MOVWF   CMCON   ; Turn off comparators
MOVLW   0x0F    ;
MOVWF   ADCON1  ; Set PORTF as digital I/O
MOVLW   0xCF    ; Value used to
                ; initialize data
                ; direction
MOVWF   TRISF   ; Set RF3:RF0 as inputs
                ; RF5:RF4 as outputs
                ; RF7:RF6 as inputs
```

 初稿

表 **9-12：** **PORTF** 功能

引脚名称	功能	TRIS 设置	I/O	缓冲器	说明
RF0/AN5/SEG18	RF0	0	O	DIG	LATF<0> 数据输出。输出不受模拟输入的影响；当使能 LCD 段输出时被禁止。
		1	I	ST	PORTF<0> 数据输入。发生 POR 时，读为 0。
	AN5	1	I	ANA	A/D 输入通道 5。发生 POR 时为默认配置。
	SEG18	x	O	ANA	LCD 的段 18 模拟输出。
RF1/AN6/C2OUT/SEG19	RF1	0	O	DIG	LATF<1> 数据输出。输出不受模拟输入的影响；当使能 LCD 段输出时被禁止。
		1	I	ST	PORTF<1> 数据输入。发生 POR 时，读为 0。
	AN6	1	I	ANA	A/D 输入通道 6。发生 POR 时为默认配置。
	C2OUT	0	O	DIG	比较器 2 输出；优先于端口数据。
	SEG19	x	O	ANA	LCD 的段 19 模拟输出。
RF2/AN7/C1OUT/SEG20	RF2	0	O	DIG	LATF<2> 数据输出。输出不受模拟输入的影响；当使能 LCD 段输出时被禁止。
		1	I	ST	PORTF<2> 数据输入。发生 POR 时，读为 0。
	AN7	1	I	ANA	A/D 输入通道 7。发生 POR 时为默认配置。
	C1OUT	0	O	TTL	比较器 1 输出；优先于端口数据。
	SEG20	x	O	ANA	LCD 的段 20 模拟输出。
RF3/AN8/SEG21	RF3	0	O	DIG	LATF<3> 数据输出。输出不受模拟输入的影响；当使能 LCD 段输出时被禁止。
		1	I	ST	PORTF<3> 数据输入。发生 POR 时，读为 0。
	AN8	1	I	ANA	A/D 输入通道 8 和比较器 C2+ 输入。发生 POR 时，默认为输入配置；不受模拟输出的影响。
	SEG21	x	O	ANA	LCD 的段 21 模拟输出。
RF4/AN9/SEG22	RF4	0	O	DIG	LATF<4> 数据输出。输出不受模拟输入的影响；当使能 LCD 段输出时被禁止。
		1	I	ST	PORTF<4> 数据输入。发生 POR 时，读为 0。
	AN9	1	I	ANA	A/D 输入通道 9 和比较器 C2- 输入。发生 POR 时，默认为输入配置；不影响数字输出。
	SEG22	x	O	ANA	LCD 的段 22 模拟输出。
RF5/AN10/CVREF/SEG23	RF5	0	O	DIG	LATF<5> 数据输出。输出不受模拟输入的影响；当使能 LCD 段或 CVREF 时被禁止。
		1	I	ST	PORTF<5> 数据输入。发生 POR 时，读为 0。
	AN10	1	I	ANA	A/D 输入通道 10 和比较器 C1+ 输入。发生 POR 时，默认为输入配置。
	CVREF	0	O	ANA	比较器参考电压输出。使能此功能会禁止数字 I/O。
	SEG23	x	O	ANA	LCD 的段 23 模拟输出。
RF6/AN11/SEG24	RF6	0	O	DIG	LATF<6> 数据输出。输出不受模拟输入的影响；当使能 LCD 段输出时被禁止。
		1	I	ST	PORTF<6> 数据输入。发生 POR 时，读为 0。
	AN11	1	I	ANA	A/D 输入通道 11 和比较器 C1- 输入。发生 POR 时，默认为输入配置；不影响数字输出。
	SEG24	x	O	ANA	LCD 的段 24 模拟输出。
RF7/$\overline{SS}$/SEG25	RF7	0	O	DIG	LATF<7> 数据输出；当使能 LCD 段输出时被禁止。
		1	I	ST	PORTF<7> 数据输入。
	$\overline{SS}$	1	I	TTL	SSP （MSSP 模块）的从动选择输入。
	SEG25	x	O	ANA	LCD 的段 25 模拟输出。

图注： PWR = 供电电源， O = 输出， I = 输入， ANA = 模拟信号， DIG = 数字输出， ST = 施密特缓冲器输入， TTL = TTL 缓冲器输入， x = 与取值无关 （TRIS 位不影响端口方向或在此可忽略）。

表 9-13： 与 PORTF 相关的寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
TRISF	PORTF 数据方向控制寄存器								62
PORTF	读 PORTF 引脚 / 写 PORTF 数据锁存器								62
LATF	读 / 写 PORTF 数据锁存器								62
ADCON1	—	—	VCFG1	VCFG0	PCFG3	PCFG2	PCFG1	PCFG0	61
CMCON	C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0	61
CVRCON	CVREN	CVROE	CVRR	CVRSS	CVR3	CVR2	CVR1	CVR0	61
LCDSE2	SE23	SE22	SE21	SE20	SE19	SE18	SE17	SE16	64
LCDSE3	SE31	SE30	SE29	SE28	SE27	SE26	SE25	SE24	64

图注： — = 未用位，读为 0。 PORTF 不使用阴影单元。

 初稿

9.7 PORTG、 TRISG 和 LATG 寄存器

PORTG 是 6 位的双向端口，对应的数据方向寄存器是 TRISG。将 TRISG 位置 1 可将对应的 PORTG 引脚配置为输入引脚（即将对应的输出驱动器置于高阻态）。将 TRISG 位清 0 可将对应的 PORTG 引脚配置为输出引脚（即输出锁存器的数据从选定引脚输出）。

数据锁存器（LATG）也是存储器映射的。对 LATG 寄存器执行读—修改—写操作将读写 PORTG 的输出锁存值。

PORTG 与 USART 和 LCD 功能复用（表 9-14）。PORTG 引脚都带有施密特触发输入缓冲器。

当使能外设功能时，应小心定义每个 PORTG 引脚的 TRIS 位。有些外设会无视 TRIS 位设置，将引脚定义为输入 / 输出引脚。用户应该查阅相应的外设章节来正确设置 TRIS 位。外设对 引脚改写值不会被装入 TRIS 寄存器。这就允许对TRIS寄存器进行读－修改－写操作，而无需考虑外设对引脚配置的修改。

PORTG<4:0> 还与由 LCDSE3 寄存器控制的 LCD 段驱动引脚复用。仅当禁止这些段时，才可使用 I/O 端口功能。

PORTG 的第 6 个引脚（$\overline{\text{MCLR}}$/VPP/RG5）只能作为输入的引脚。其操作由 MCLRE 配置位控制。当被选定为端口引脚（MCLRE = 0）时，它充当数字信号输入引脚；因此，其操作与 TRIS 或 LAT 位无关。否则，它充当器件的主清零输入。对于任一配置，RG5 均充当编程过程中的编程电压输入引脚。

注：	在上电复位时，仅当主清零功能被禁止时，才将 RG5 使能为数字输入。而其他的 5 个引脚均被配置为数字输入。

例 9-7： 初始化 PORTG

```
CLRF    PORTG   ; Initialize PORTG by
                ; clearing output
                ; data latches
CLRF    LATG    ; Alternate method
                ; to clear output
                ; data latches
MOVLW   0x04    ; Value used to
                ; initialize data
                ; direction
MOVWF   TRISG   ; Set RG1:RG0 as outputs
                ; RG2 as input
                ; RG4:RG3 as inputs
```

表 9-14： PORTG 功能

引脚名称	功能	TRIS 设置	I/O	缓冲器	说明
RG0/SEG30	RG0	0	O	DIG	LATG<0> 数据输出；当使能 LCD 段输出时被禁止。
		1	I	ST	PORTG<0> 数据输入。
	SEG30	x	O	ANA	LCD 的段 30 模拟输出。
RG1/TX2/CK2/SEG29	RG1	0	O	DIG	LATG<1> 数据输出；当使能 LCD 段输出时被禁止。
		1	I	ST	PORTG<1> 数据输入。
	TX2	1	O	DIG	同步串行数据输出 （AUSART 模块）；优先于端口数据。
	CK2	1	O	DIG	同步串行数据输入 （AUSART 模块）。用户必须将其配置为输入。
		1	I	ST	同步串行时钟输入 （AUSART 模块）。
	SEG29	x	O	ANA	LCD 的段 29 模拟输出。
RG2/RX2/DT2/SEG28	RG2	0	O	DIG	LATG<2> 数据输出；当使能 LCD 段输出时被禁止。
		1	I	ST	PORTG<2> 数据输入。
	RX2	1	I	ST	异步串行接收数据输入 （AUSART 模块）。
	DT2	1	O	DIG	同步串行数据输出 （AUSART 模块）；优先于端口数据。
		1	I	ST	同步串行数据输入 （AUSART 模块）。用户必须将其配置为输入。
	SEG28	x	O	ANA	LCD 的段 28 模拟输出。
RG3/SEG27	RG3	0	O	DIG	LATG<3> 数据输出；当使能 LCD 段输出时被禁止。
		1	I	ST	PORTG<3> 数据输入。
	SEG27	0	O	ANA	LCD 的段 27 模拟输出。
RG4/SEG26	RG4	0	O	DIG	LATG<4> 数据输出；当使能 LCD 段输出时被禁止。
		1	I	ST	PORTG<4> 数据输入。
	SEG26	x	O	ANA	LCD 的段 26 模拟输出。
$\overline{\text{MCLR}}$/VPP/RG5	$\overline{\text{MCLR}}$	—(1)	I	ST	外部主清零输入；当 MCLRE 配置位置 1 时使能。
	VPP	—(1)	I	ANA	高压检测；用于 ICSP™ 模式的入口检测。无论引脚处于何种模式，它总是可用的。
	RG5	—(1)	I	ST	PORTG<5> 数据输入；当 MCLRE 配置位清零时使能。

图注： PWR = 供电电源， O = 输出， I = 输入， ANA = 模拟信号， DIG = 数字输出， ST = 施密特缓冲器输入，TTL = TTL 缓冲器输入， x = 与取值无关 （TRIS 位不影响端口方向或在此可忽略）。

注 1： RG5 没有对应的 TRISG 位。

表 9-15： 与 PORTG 相关的寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
PORTG	—	—	RG5(1)	读 PORTG 引脚 / 写 PORTG 数据锁存器					62
LATG	—	—	—	LATG 数据输出寄存器					62
TRISG	—	—	—	PORTG 的数据方向控制寄存器					62
LCDSE3	SE31	SE30	SE29	SE28	SE27	SE26	SE25	SE24	64

图注： — = 未用，读为 0。 PORTG 不使用阴影单元。

注 1： 仅当禁止 $\overline{\text{MCLR}}$ 时，才可将 RG5 用作输入。

9.8 PORTH、 LATH 和 TRISH 寄存器

注：	只有 80 引脚的器件上才有 PORTH。

PORTH 是 8 位的双向 I/O 端口，对应的数据方向寄存器是 TRISH。将 TRISH 位置 1 可将对应的 PORTH 引脚配置为输入引脚（即将对应的输出驱动器置于高阻态）。将 TRISH 位清 0 可将对应的 PORTH 引脚配置为输出引脚（即输出锁存器的数据从选定引脚输出）。

数据锁存器（LATH）也是存储器映射的。对 LATH 寄存器执行读—修改—写操作将读写 PORTH 的输出锁存值。

PORTH 上所有引脚都配有施密特触发输入缓冲器。每个引脚都可被单独地配置为输入或输出。

注：	在上电复位时，这些引脚被配置为数字输入。

PORTH 还与由 LCDSE5 寄存器控制的 LCD 段驱动引脚复用。仅当禁止这些段时，才可使用 I/O 端口功能。

例 9-8： 初始化 PORTH

```
CLRF    PORTH       ; Initialize PORTH by
                    ; clearing output
                    ; data latches
CLRF    LATH        ; Alternate method
                    ; to clear output
                    ; data latches
MOVLW   0CFh        ; Value used to
                    ; initialize data
                    ; direction
MOVWF   TRISH       ; Set RH3:RH0 as inputs
                    ; RH5:RH4 as outputs
                    ; RH7:RH6 as inputs
```

表 9-16： PORTH 功能

引脚名称	功能	TRIS 设置	I/O	缓冲器	说明
RH0/SEG47	RH0	0	O	DIG-4	LATH<0> 数据输出；当使能 LCD 段输出时被禁止。
		1	I	ST	PORTH<0> 数据输入。
	SEG47	x	O	ANA	LCD 的段 47 模拟输出。
RH1/SEG46	RH1	0	O	DIG	LATH<1> 数据输出；当使能 LCD 段输出时被禁止。
		1	I	ST	PORTH<1> 数据输入。
	SEG46	x	O	ANA	LCD 的段 46 模拟输出。
RH2/SEG45	RH2	0	O	DIG	LATH<2> 数据输出；当使能 LCD 段输出时被禁止。
		1	I	ST	PORTH<2> 数据输入。
	SEG45	x	O	ANA	LCD 的段 45 模拟输出。
RH3/SEG44	RH3	0	O	DIG	LATH<3> 数据输出；当使能 LCD 段输出时被禁止。
		1	I	ST	PORTH<3> 数据输入。
	SEG44	x	O	ANA	LCD 的段 44 模拟输出。
RH4/SEG40	RH4	0	O	DIG	LATH<4> 数据输出；当使能 LCD 段输出时被禁止。
		1	I	ST	PORTH<4> 数据输入。
	SEG40	x	O	ANA	LCD 的段 40 模拟输出。
RH5/SEG41	RH5	0	O	DIG	LATH<5> 数据输出；当使能 LCD 段输出时被禁止。
		1	I	ST	PORTH<5> 数据输入。
	SEG41	x	O	ANA	LCD 的段 41 模拟输出。
RH6/SEG42	RH6	0	O	DIG	LATH<6> 数据输出；当使能 LCD 段输出时被禁止。
		1	I	ST	PORTH<6> 数据输入。
	SEG42	x	O	ANA	LCD 的段 42 模拟输出。
RH7/SEG43	RH7	0	O	DIG	LATH<7> 数据输出；当使能 LCD 段输出时被禁止。
		1	I	ST	PORTH<7> 数据输入。
	SEG43	x	O	ANA	LCD 的段 43 模拟输出。

图注： PWR = 供电电源， O = 输出， I = 输入， ANA = 模拟信号， DIG = 数字输出， ST = 施密特缓冲器输入，
TTL = TTL 缓冲器输入， x = 与取值无关 （TRIS 位不影响端口方向或在此可忽略）。

表 9-17： 与 PORTH 相关的寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
TRISH	PORTH 数据方向控制寄存器								62
PORTH	读 PORTH 引脚 / 写 PORTH 数据锁存器								62
LATH	读 / 写 PORTH 数据锁存器								62
LCDSE5	SE47	SE46	SE45	SE44	SE43	SE42	SE41	SE40	64

9.9 PORTJ、 TRISJ 和 LATJ 寄存器

注：	只有 80 引脚的器件上才有 PORTJ。

PORTJ 是 8 位的双向端口，对应的数据方向寄存器是 TRISJ。将 TRISJ 位置 1 可将对应的 PORTJ 引脚配置为输入引脚 （即将对应的输出驱动器置于高阻态）。将 TRISJ 位清 0 可将对应的 PORTJ 引脚配置为输出引脚（即输出锁存器的数据从选定引脚输出）。

数据锁存器（LATJ）也是存储器映射的。对 LATJ 寄存器执行读—修改—写操作将读写 PORTJ 的输出锁存值。

PORTJ 上所有引脚均具有由施密特触发输入缓冲器。每个引脚都可被单独地配置为输入或输出。

注：	在上电复位时，这些引脚被配置为数字输入。

PORTJ 还与由 LCDSE4 寄存器控制的 LCD 段驱动引脚复用。仅当禁止这些段时，才可使用 I/O 端口功能。

例 9-9： 初始化 PORTJ

```
CLRF   PORTJ   ; Initialize PORTG by
               ; clearing output
               ; data latches
CLRF   LATJ    ; Alternate method
               ; to clear output
               ; data latches
MOVLW 0xCF     ; Value used to
               ; initialize data
               ; direction
MOVWF TRISJ    ; Set RJ3:RJ0 as inputs
               ; RJ5:RJ4 as output
               ; RJ7:RJ6 as inputs
```

表 9-18： PORTJ 功能

引脚名称	功能	TRIS 设置	I/O	缓冲器	说明
RJ0/SEG32	RJ0	0	O	DIG	LATJ<0> 数据输出；当使能 LCD 段输出时被禁止。
		1	I	ST	PORTJ<0> 数据输入。
	SEG32	x	O	ANA	LCD 的段 32 模拟输出。
RJ1/SEG33	RJ1	0	O	DIG	LATJ<1> 数据输出；当使能 LCD 段输出时被禁止。
		1	I	ST	PORTJ<1> 数据输入。
	SEG33	x	O	ANA	LCD 的段 33 模拟输出。
RJ2/SEG34	RJ2	0	O	DIG	LATJ<2> 数据输出；当使能 LCD 段输出时被禁止。
		1	I	ST	PORTJ<2> 数据输入。
	SEG34	x	O	ANA	LCD 的段 34 模拟输出。
RJ3/SEG35	RJ3	0	O	DIG	LATJ<3> 数据输出；当使能 LCD 段输出时被禁止。
		1	I	ST	PORTJ<3> 数据输入。
	SEG35	x	O	ANA	LCD 的段 35 模拟输出。
RJ4/SEG39	RJ4	0	O	DIG	LATJ<4> 数据输出；当使能 LCD 段输出时被禁止。
		1	I	ST	PORTJ<4> 数据输入。
	SEG39	x	O	ANA	LCD 的段 39 模拟输出。
RJ5/SEG38	RJ5	0	O	DIG	LATJ<5> 数据输出；当使能 LCD 段输出时被禁止。
		1	I	ST	PORTJ<5> 数据输入。
	SEG38	x	O	ANA	LCD 的段 38 模拟输出。
RJ6/SEG37	RJ6	0	O	DIG	LATJ<6> 数据输出；当使能 LCD 段输出时被禁止。
		1	I	ST	PORTJ<6> 数据输入。
	SEG37	x	O	ANA	LCD 的段 37 模拟输出。
RJ7/SEG36	RJ7	0	O	DIG	LATJ<7> 数据输出；当使能 LCD 段输出时被禁止。
		1	I	ST	PORTJ<7> 数据输入。
	SEG36	x	O	ANA	LCD 的段 36 模拟输出。

图注： PWR = 供电电源， O = 输出， I = 输入， ANA = 模拟信号， DIG = 数字输出， ST = 施密特缓冲器输入， TTL = TTL 缓冲器输入， x = 与取值无关 （TRIS 位不影响端口方向或在此可忽略）。

表 9-19： 与 PORTJ 相关的寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
PORTJ	读 PORTJ 引脚 / 写 PORTJ 数据锁存器								62
LATJ	LATJ 数据输出寄存器								62
TRISJ	PORTJ 的数据方向控制寄存器								62
LCDSE4	SE39	SE38	SE37	SE36	SE35	SE34	SE33	SE32	64

 初稿

10.0 TIMER0 模块

Timer0 模块具有以下特征：

- 可通过软件选择，作为 8 位或 16 位定时器 / 计数器
- 可读写寄存器
- 专用的 8 位软件可编程预分频器
- 可选的时钟源 （内部或外部）
- 外部时钟的边沿选择
- 溢出时中断

T0CON 寄存器 （寄存器 10-1）控制该模块的工作方式，包括预分频比值的选择。该寄存器是可读写的。

图 10-1 给出了 8 位模式下 Timer0 模块的简化框图，图 10-2 给出了 16 位模式下 Timer0 模块的简化框图。

寄存器 10-1： T0CON：TIMER0 控制寄存器

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
TMR0ON	T08BIT	T0CS	T0SE	PSA	T0PS2	T0PS1	T0PS0
bit 7							bit 0

bit7 **TMR0ON**：Timer0 开关控制位
1 = 使能 Timer0
0 = 禁止 Timer0

bit6 **T08BIT**：Timer0 8 位 /16 位控制位
1 = Timer0 被配置为 8 位定时器 / 计数器
0 = Timer0 被配置为 16 位定时器 / 计数器

bit5 **T0CS**：Timer0 时钟源选择位
1 = T0CKI 引脚上的传输信号作为时钟源
0 = 内部指令周期时钟 （CLKO）作为时钟源

bit4 **T0SE**：Timer0 时钟源边沿选择位
1 = 在 T0CKI 引脚上电平的下降沿递增
0 = 在 T0CKI 引脚上电平的上升沿递增

bit3 **PSA**：Timer0 预分频器分配位
1 = 未分配 Timer0 预分频器。 Timer0 时钟输入不经过预分频器
0 = 已分配 Timer0 预分频器。 Timer0 时钟输入来自预分频器的输出

bit2-0 **T0PS2:T0PS0**：Timer0 预分频值选择位
111 = 1:256 预分频值
110 = 1:128 预分频值
101 = 1:64 预分频值
100 = 1:32 预分频值
011 = 1:16 预分频值
010 = 1:8 预分频值
001 = 1:4 预分频值
000 = 1:2 预分频值

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

10.1 Timer0 工作原理

Timer0 既可用作定时器亦可用作计数器。将 T0CS 位（T0CON<5>）清零即可选择定时器模式。在定时器模式下（T0CS = 0），Timer0 模块在每个时钟周期计时都会递增（默认情况下），除非选择了其他预分频比值（见第 **10.3 节“预分频器”**）。如果写入 TMR0 寄存器，那么在随后的两个指令周期内，计时都不再递增。用户可通过将校正值写入 TMR0 寄存器来解决上述问题。

通过将T0CS位置1选择计数器模式。在计数器模式下，Timer0 可在 RA4/T0CKI 引脚上电平的每个上升沿或下降沿递增。触发递增的边沿由 Timer0 时钟源边沿选择位 T0SE（T0CON<4>）决定。清零此位即选择上升沿。下面讨论外部时钟输入的限制条件。

可以使用外部时钟源来驱动 Timer0。但是，必须确保外部时钟和内部时钟相位（TOSC）同步。在同步之后，定时器 / 计数器仍需要一定的延时才会引发递增操作。

10.2 Timer0 的 16 位读写模式

TMR0H 并不是 16 位模式下 Timer0 的高字节，而是被缓存的 Timer0 高字节，不可以被直接读写（见图 10-2）。在读 TMR0L 时使用 Timer0 高字节的内容更新 TMR0H。这样可以一次读取 Timer0 的全部 16 位，而无需验证读到的高字节和低字节的有效性（在连续读取高字节和低字节时，由于可能存在进位，因此需要验证读到的高字节和低字节的有效性 ）。

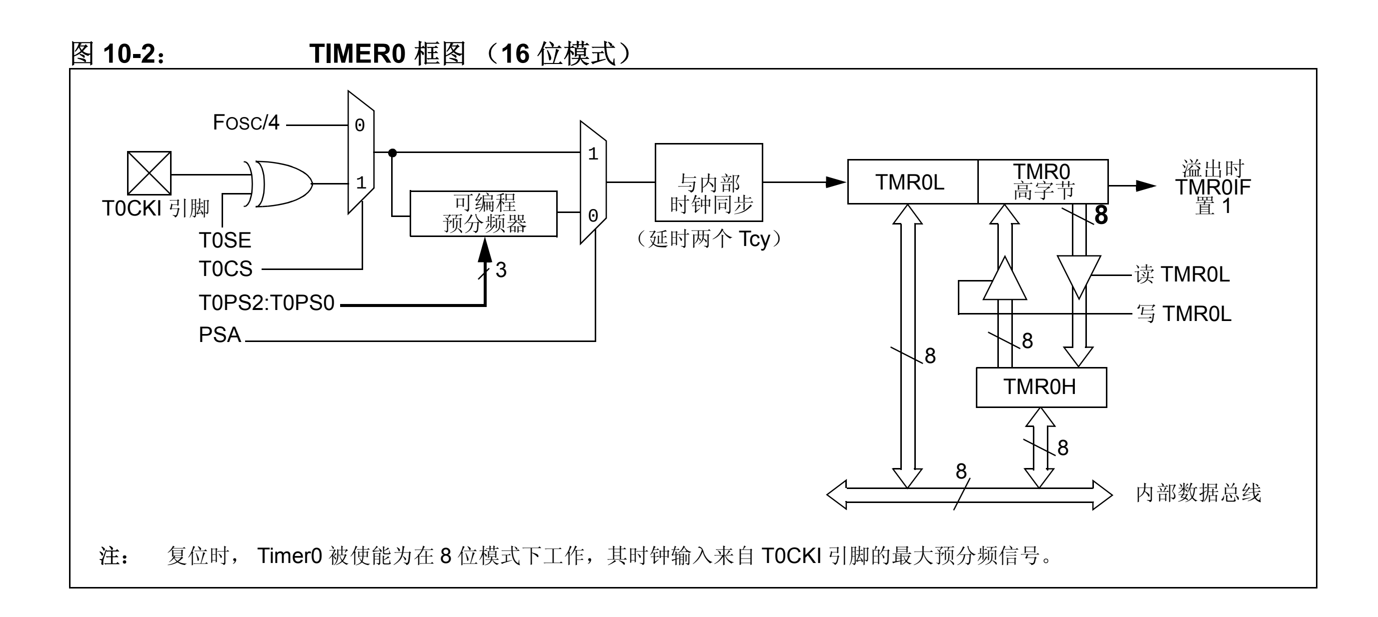
同样，写入 Timer0 的高字节也是通过 TMR0H 缓冲寄存器来操作的。在写入 TMR0L 的同时，使用 TMR0H 的内容更新 Timer0 的高字节。这样一次就可以完成 Timer0 全部 16 位的更新。

图 10-1： TIMER0 框图 （8 位模式）

FOSC/4
T0CKI 引脚
T0SE
T0CS
T0PS2:T0PS0
PSA
0
1
可编程
预分频器
3
1
0
与内部
时钟同步
（延时两个 TCY）
TMR0L
溢出时
TMR0IF
置 1
8
8
内部数据总线

注： 复位时， Timer0 被使能为在 8 位模式下工作，其时钟输入来自 T0CKI 引脚的最大预分频信号。

图 10-2： TIMER0 框图 （16 位模式）



注： 复位时， Timer0 被使能为在 8 位模式下工作，其时钟输入来自 T0CKI 引脚的最大预分频信号。

10.3 预分频器

Timer0 模块的预分频器为一个 8 位计数器。该预分频器不可直接读写。通过对 PSA 和 T0PS2:T0PS0（T0CON<3:0>）位设置，来确定预分频器的分配和预分频比值。

将 PSA 位清零可将预分频器分配给 Timer0 模块。预分频比值可以在 1:2 到 1:256 之间进行选择，以 2 的整数次幂递增。

若将预分频器分配给 Timer0 模块，所有以 TMR0 寄存器为写入对象的指令（如 `CLRF TMR0`、`MOVWF TMR0` 和 `BSF TMR0` 等）都将使预分频器的计数值清零。

> **注：** 若将预分频器分配给 Timer0，写入 TMR0 会将预分频器的计数值清零，但不会改变预分频器的分配。

10.3.1 切换预分频器的分配

预分频器的分配完全由软件控制，并且在程序执行期间可以随时更改。

10.4 Timer0 中断

当 TMR0 寄存器发生溢出时，将产生 TMR0 中断。这种溢出会使 TMR0IF 标志位置 1。清零 TMR0IE 位（INTCON<5>）可屏蔽此中断。在重新使能该中断前，必须在中断服务程序中用软件清零 TMR0IF 位。

由于 Timer0 在休眠模式下是关闭的，所以 TMR0 中断无法将处理器从休眠状态唤醒。

表 10-1： 与 TIMER0 相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
TMR0L	Timer0 模块的低字节寄存器								60
TMR0H	Timer0 模块的高字节寄存器								60
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
T0CON	TMR0ON	T08BIT	T0CS	T0SE	PSA	T0PS2	T0PS1	T0PS0	60
TRISA	PORTA 数据方向寄存器								62

图注： 在 Timer0 模块中未使用阴影单元。

注：

11.0 TIMER1 模块

Timer1 定时器 / 计数器模块具有以下特征：

- 可通过软件选择，作为 16 位定时器或计数器
- 可读写的 8 位寄存器 （TMR1H 和 TMR1L）
- 可选择使用器件时钟或 Timer1 内部振荡器作为时钟源
- 溢出时中断
- 在触发 CCP 特殊事件时复位
- 器件时钟状态标志位 （T1RUN）

图 11-1 所示为 Timer1 模块的简化框图。图 11-2 所示为此模块在读写模式下的工作原理框图。

此模块具有低功耗振荡器可提供额外的时钟。Timer1 振荡器也可作为单片机处于节能状态时的低功耗时钟源。

在对外部元件数量和代码开销要求苛刻的应用中，Timer1 可以为其提供实时时钟 （RTC）。

Timer1 由 T1CON 控制寄存器 （寄存器 11-1）控制。该寄存器还有 Timer1 振荡器使能位（T1OSCEN）。可以通过将控制位 TMR1ON （T1CON<0>）置 1 或清零来使能或禁止 Timer1。

寄存器 11-1： T1CON：TIMER1 控制寄存器

R/W-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
RD16	T1RUN	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{\text{T1SYNC}}$	TMR1CS	TMR1ON
bit 7							bit 0

bit 7 **RD16：**16 位读 / 写模式使能位
1 = 使能 Timer1 通过一次 16 位操作进行寄存器读 / 写
0 = 使能 Timer1 通过两次 8 位操作进行寄存器读 / 写

bit 6 **T1RUN：**Timer1 系统时钟状态位
1 = 器件时钟由 Timer1 振荡器产生
0 = 器件时钟由另一个时钟源产生

bit 5-4 **T1CKPS1:T1CKPS0：**Timer1 输入时钟预分频值选择位
11 = 1:8 预分频值
10 = 1:4 预分频值
01 = 1:2 预分频值
00 = 1:1 预分频值

bit 3 **T1OSCEN：**Timer1 振荡器使能位
1 = 使能 Timer1 振荡器
0 = 关闭 Timer1 振荡器
关断振荡器的反相器和反馈电阻以降低功耗。

bit 2 **$\overline{\text{T1SYNC}}$：**Timer1 外部时钟输入同步选择位
当 TMR1CS = 1 时：
1 = 不同步外部时钟输入
0 = 同步外部时钟输入
当 TMR1CS = 0 时：
忽略此位。当 TMR1CS = 0 时， Timer1 使用内部时钟。

bit 1 **TMR1CS：**Timer1 时钟源选择位
1 = 使用 RC0/T1OSO/T13CKI 引脚上的信号作为外部时钟 （上升沿计数）
0 = 内部时钟 （Fosc/4）

bit 0 **TMR1ON：**Timer1 使能位
1 = 使能 Timer1
0 = 停止 Timer1

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

11.1 Timer1 工作原理

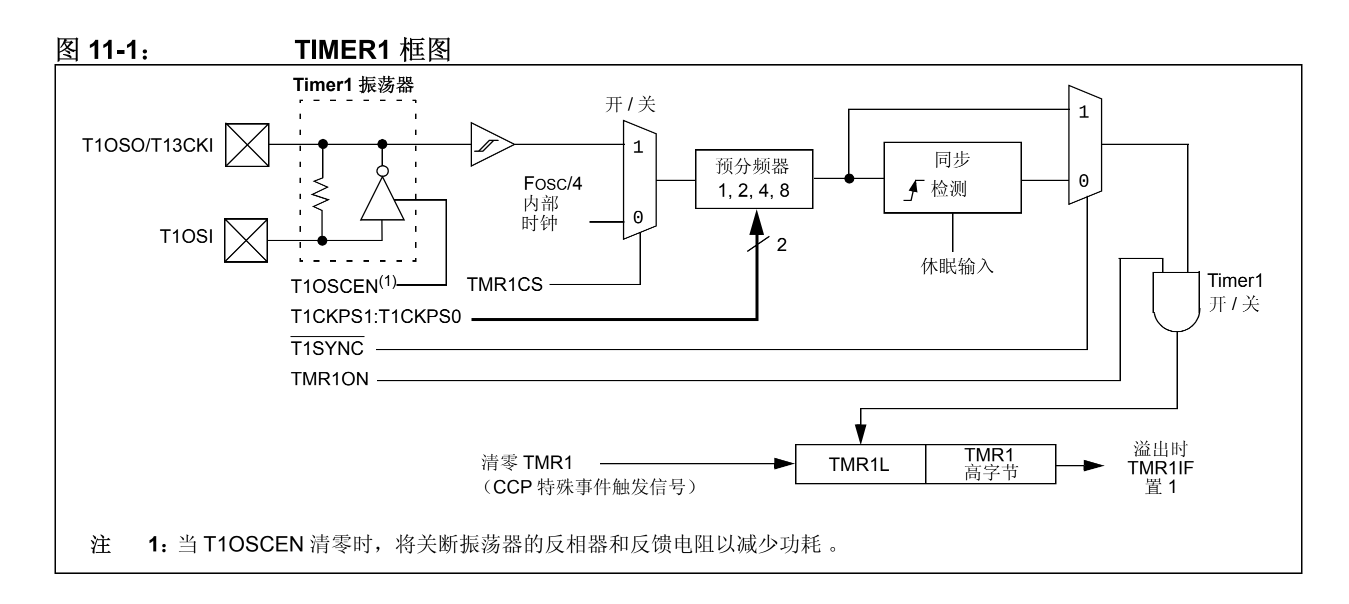
Timer1 可在以下模式工作：

- 定时器
- 同步计数器
- 异步计数器

工作模式由时钟选择位 TMR1CS（T1CON<1>）决定。当 TMR1CS 清零 （ = 0）时， Timer1 在每个内部指令周期 （Fosc/4）计时 / 计数递增。当 TMR1CS 位置 1 时，Timer1 在 Timer1 外部时钟输入信号或 Timer1 振荡器输出信号 （如果使能）的每个上升沿计时 / 计数递增。

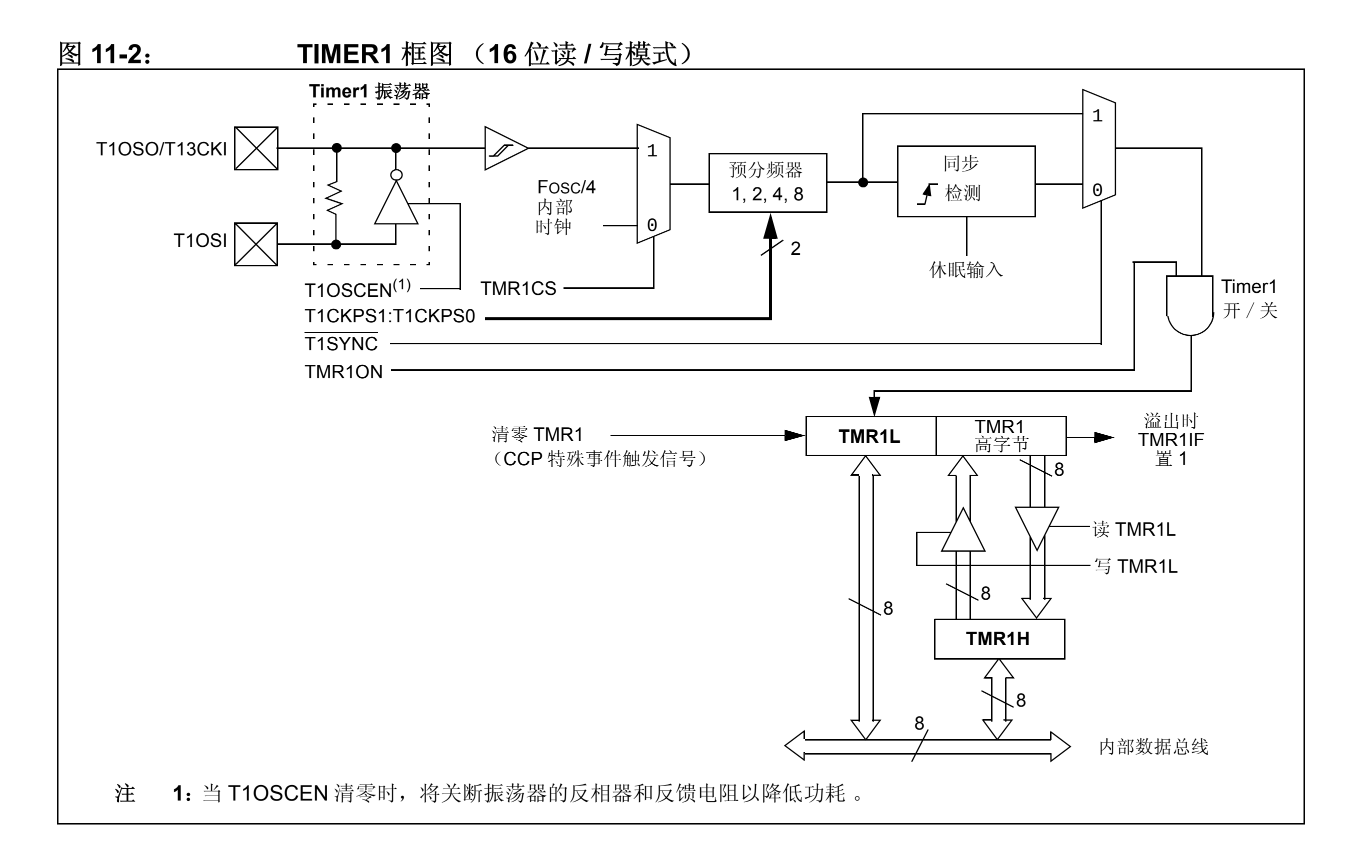
当使能 Timer1 时，RC1/T1OSI 和 RC0/T1OSO/T13CKI 引脚变为输入引脚。这意味着 TRISC<1:0> 的值被忽略并且这些引脚将读为 0。

图 11-1： TIMER1 框图



注 1：当 T1OSCEN 清零时，将关断振荡器的反相器和反馈电阻以减少功耗 。

图 11-2： TIMER1 框图 （16 位读 / 写模式）



注 1：当 T1OSCEN 清零时，将关断振荡器的反相器和反馈电阻以降低功耗 。

 初稿

11.2 Timer1 的 16 位读 / 写模式

可将 Timer1 配置为 16 位读写模式（见图 11-2）。当 RD16 控制位（T1CON<7>）置 1 时， TMR1H 的地址被映射到 Timer1 的高字节缓冲寄存器。对 TMR1L 的读操作将把 Timer1 的高字节装入 Timer1 高字节缓冲寄存器。这种方式使用户可以精确地读取 Time1 的全部 16 位，而不需要像先读高字节再读低字节那样由于两次读取之间可能存在进位，而不得不验证读取的有效性。

对 Timer1 的高字节进行写操作也必须通过 TMR1H 缓冲寄存器进行。在写入 TMR1L 的同时，使用 TMR1H 的内容更新 Timer1 的高字节。这样允许用户将 16 位值一次写入 Timer1 的高字节和低字节。

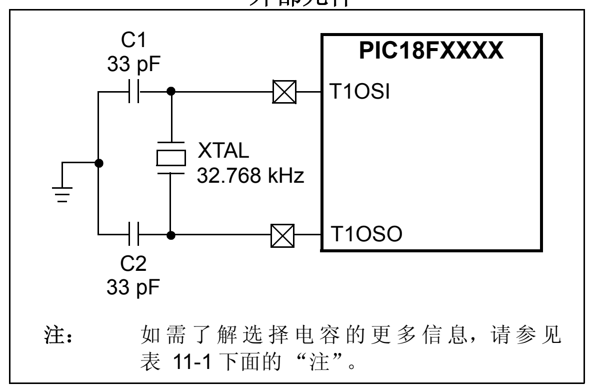
在该模式下不能直接读写 Timer1 的高字节，所有读写都必须通过 Timer1 高字节缓冲寄存器来进行。写入 TMR1H 不会清零 Timer1 预分频器，只有在写 TMR1L 时才会清零该预分频器。

11.3 Timer1 振荡器

片上晶体振荡器电路连接在 T1OSI （输入）引脚和 T1OSO （放大器输出）引脚之间。可以通过将 Timer1 振荡器使能位 T1OSCEN （T1CON<3>）置 1 来使能该振荡电路。此振荡电路是一种低功耗电路，它采用了额定振荡频率为 32 kHz 的晶振，在所有的节能模式下都可继续运行。图 11-3所示是典型的LP振荡器电路。表 11-1 给出了供 Timer1 振荡器选择的电容值。

用户必须提供软件延迟来确保 Timer1 振荡器的正常起振。

图 11-3： TIMER1 LP振荡器的外部元件



注： 如需了解选择电容的更多信息，请参见表 11-1 下面的 “注”。

表 11-1： TIMER1 振荡器（2-4）的电容选择

振荡类型	频率	C1	C2
LP	32 kHz	27 pF(1)	27 pF(1)

注 1：Microchip 建议在验证振荡器电路时这些值仅供参考。
2：选用较大的电容值虽然可以提高振荡器的稳定性，但同时也会延长起振时间。
3：由于谐振器 / 晶振的特性各不相同，因此用户应当向谐振器 / 晶振制造厂商咨询外围元件的相应值。
4：上述电容值仅供设计参考。

11.3.1 使用 TIMER1 作为时钟源

在节能模式中也可以将 Timer1 振荡器用作时钟源。通过将时钟选择位 SCS1:SCS0 （OSCCON<1:0>）设置为 “01”，器件可以切换到 SEC_RUN 模式， CPU 和外设都可以用 Timer1 振荡器作为时钟源。如果 IDLEN 位 （OSCCON<7>）被清零并且执行了 SLEEP 指令，器件将进入 SEC_IDLE 模式。欲知更多详情，请参见**第 3.0 节 “功耗管理模式”**。

无论何时将 Timer1 振荡器用作时钟源， Timer1 系统时钟状态标志位 T1RUN （T1CON<6>）均会置 1。这可用于确定控制器的当前时钟模式。该位也可指示故障保护时钟监视器当前正使用的时钟源。如果使能了故障保护时钟监视器并且 Timer1 振荡器在提供时钟信号时发生了故障，轮询 T1RUN 位可以确定时钟源是 Timer1 振荡器还是其他时钟源。

11.3.2 低功耗 TIMER1 选项

根据器件配置，Timer1 振荡器可以在两种不同的功耗级别下工作。当 LPT1OSC 配置位置 1 时，Timer1 振荡器在低功耗模式下工作。当 LPT1OSC 清零时，Timer1 在高功耗模式下工作。不管器件工作在什么模式下，特定模式的功耗都是相对固定的。默认将 Timer1 配置为工作于功耗较高的模式下。

由于低功耗模式对干扰更加敏感，噪声环境可能会导致某些振荡器不稳定。因此低功耗选项最适合那些需要重点考虑节省功耗的低噪声应用。

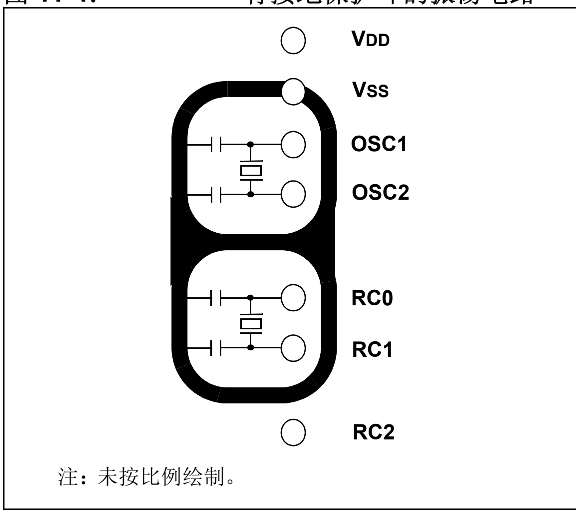
11.3.3 TIMER1 振荡器布线注意事项

Timer1 振荡器在工作期间仅消耗极少的电流。鉴于此振荡器的低功耗特性，它对附近变化较快的信号比较敏感。

如图 11-3所示，振荡电路应该尽可能靠近单片机。除了 VSS 或 VDD 外，在该振荡电路区域的不应有其他电路。

如果必须要在该振荡器附近布置高速电路（如输出比较模式或 PWM 模式的 CCP1 引脚，或使用 OSC2 引脚的主振荡器），那么在该振荡电路周围布置接地保护环（如图 11-4 所示），对于单面 PCB 板或外加接地层的电路板来讲可能会有帮助。

图 11-4： 有接地保护环的振荡电路



11.4 Timer1 中断

TMR1 寄存器对（TMR1H:TMR1L）从 0000h 开始加 1，一直到 FFFFh，然后溢出从 0000h 重新开始计数。如果使能了 Timer1 中断，该中断就会在溢出时产生，并由中断标志位 TMR1IF （PIR1<0>）表示。可以通过对 Timer1 中断使能位 TMR1IE（PIE1<0>）置 1 或清零来使能或禁止该中断。

11.5 使用 CCP 特殊事件触发信号来复位 Timer1

如果 CCP 模块被配置为产生特殊事件触发信号（CCP1M3:CCP1M0 或 CCP2M3:CCP2M0 = `1011`）的比较模式，该触发信号将复位 Timer1。如果使能了 A/D 模块，来自 CCP2 的触发信号还将启动 A/D 转换（欲知更多信息，请参见**第 14.3.4 节 “特殊事件触发器”**。）。

要使用这一功能，必须将 Timer1 配置为定时器或同步计数器。在这种情况下，CCPRH:CCPRL这对寄存器实际上变成了 Timer1 的周期寄存器。

如果 Timer1 在异步计数器模式下运行，复位操作可能不起作用。

如果 Timer1 的写操作和特殊事件触发同时发生，则写操作优先。

> **注：** 来自 CCP2 模块的特殊事件触发信号不会将中断标志位 TMR1IF（PIR1<0>）置 1。

11.6 使用 Timer1 作为实时时钟

为 Timer1 添加外部 LP 振荡器（如**第 11.3 节“Timer1 振荡器”**中所述），可以为用户提供 RTC 功能。这是通过一个提供精确时基的廉价时钟晶振以及几行计算时间的应用程序代码实现的。当器件在休眠模式下工作并使用电池或超大容量电容作为电源时，可省去另外的 RTC 器件和备用电池。

应用代码程序 `RTCisr`（如例 11-1 所示），演示了使用中断服务程序以 1 秒的间隔递增计数器的简单方法。将 TMR1 寄存器对的值加 1 直至溢出将触发中断并调用中断服务程序，该程序会使秒计数器加 1，其他的分钟和小时计数器则会在前面的计数器溢出时加 1。

由于这对寄存器为 16 位宽，因此使用 32.768 kHz 时钟，将其计数到溢出需要 2 秒。要使溢出按所需的 1 秒间隔进行，必须预先装载这对寄存器。最简单的方法是使用 BSF 指令将 TMR1H 的最高有效位置 1。请注意决不要预先加载或改变 TMR1L 寄存器，这样做可能会引起多个周期的累积错误。

要使此方法精确，Timer1 必须工作于异步模式且必须使能 Timer1 溢出中断（PIE1<0> = `1`），如程序 `RTCinit` 所示。同时 Timer1 振荡器也必须使能并始终保持运行。

例 11-1： 使用 TIMER1 中断服务实现实时时钟

```
RTCinit
            MOVLW     80h              ; Preload TMR1 register pair
            MOVWF     TMR1H            ; for 1 second overflow
            CLRF      TMR1L
            MOVLW     b&#xD5;00001111&#xD5;; Configure for external clock,
            MOVWF     T1OSC            ; Asynchronous operation, external oscillator
            CLRF      secs             ; Initialize timekeeping registers
            CLRF      mins             ;
            MOVLW     .12
            MOVWF     hours
            BSF       PIE1, TMR1IE     ; Enable Timer1 interrupt
            RETURN
RTCisr
            BSF       TMR1H, 7         ; Preload for 1 sec overflow
            BCF       PIR1, TMR1IF     ; Clear interrupt flag
            INCF      secs, F          ; Increment seconds
            MOVLW     .59              ; 60 seconds elapsed?
            CPFSGT    secs
            RETURN                     ; No, done
            CLRF      secs             ; Clear seconds
            INCF      mins, F          ; Increment minutes
            MOVLW     .59              ; 60 minutes elapsed?
            CPFSGT    mins
            RETURN                     ; No, done
            CLRF      mins             ; clear minutes
            INCF      hours, F         ; Increment hours
            MOVLW     .23              ; 24 hours elapsed?
            CPFSGT    hours
            RETURN                     ; No, done
            MOVLW     .01              ; Reset hours to 1
            MOVWF     hours
            RETURN                     ; Done
```

表 11-2： TIMER1 作为定时器 / 计数器时相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
PIR1	—	ADIF	RC1IF	TX1IF	SSPIF	CCP1IF	TMR2IF	TMR1IF	61
PIE1	—	ADIE	RC1IE	TX1IE	SSPIE	CCP1IE	TMR2IE	TMR1IE	61
IPR1	—	ADIP	RC1IP	TX1IP	SSPIP	CCP1IP	TMR2IP	TMR1IP	61
TMR1L	16 位 TMR1 寄存器的低 8 位保持寄存器								60
TMR1H	16 位 TMR1 寄存器的高 8 位保持寄存器								60
T1CON	RD16	T1RUN	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{\text{T1SYNC}}$	TMR1CS	TMR1ON	60

图注： — = 未用位，读为 0。在 Timer1 模块中未使用阴影单元。

注：

12.0 TIMER2 模块

Timer2 定时器模块具有以下特征：

- 8 位定时器和周期寄存器 （分别为 TMR2 和 PR2）
- 可读写 （以上两个寄存器）
- 可软件编程的预分频器 （分频比为 1:1、 1:4 和 1:16）
- 可软件编程的后分频器 （分频比为 1:1 到 1:16）
- 当 TMR2 与 PR2 匹配时中断
- 作为 MSSP 模块的可选移位时钟

此模块由 T2CON 寄存器（寄存器 12-1）控制，此寄存器使能或禁止定时器并配置预分频器和后分频器。可以通过清零控制位 TMR2ON（T2CON<2>）关闭 Timer2，以实现功耗最小。

图 12-1 所示为此模块的简化框图。

12.1 Timer2 工作原理

在正常情形下， TMR2 从 00h 开始加 1，每一时钟周期（Fosc/4）计数一次。2 位计数器 / 预分频器提供了对时钟输入不分频、4 分频和 16 分频三种预分频选项，并可通过预分频控制位，T2CKPS1:T2CKPS0（T2CON<1:0>）进行选择。在每个时钟周期， TMR2 的值都会与周期寄存器PR2中的值进行比较。当两个值匹配时，由比较器产生匹配信号作为Timer2的输出。此信号也会使 TMR2 的值在下一个周期复位到 00h，并驱动输出计数器 / 后分频器 （见第 **12.2 节 “Timer2 中断”**）。

TMR2 和 PR2 寄存器均可直接读写。在任何器件复位时，TMR2 寄存器都会清零，而 PR2 寄存器则初始化为 FFh。预分频和后分频计数器均会在发生以下事件时清零：

- 对 TMR2 寄存器进行写操作
- 对 T2CON 寄存器进行写操作
- 任何方式的器件复位 （上电复位、 $\overline{\text{MCLR}}$ 复位、看门狗定时器复位或者欠压复位）

写 T2CON 时 TMR2 不会清零。

寄存器 12-1： T2CON：TIMER2 控制寄存器

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0
bit 7							bit 0

bit 7　未用位：读为 0

bit 6-3　**T2OUTPS3:T2OUTPS0**：Timer2 输出后分频比值选择位

0000 = 1:1 后分频比
0001 = 1:2 后分频比
•
•
•
1111 = 1:16 后分频比

bit 2　**TMR2ON**：Timer2 使能位

1 = 使能 Timer2
0 = 关闭 Timer2

bit 1-0　**T2CKPS1:T2CKPS0**：Timer2 时钟预分频值选择位

00 = 预分频值为 1
01 = 预分频值为 4
1x = 预分频值为 16

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

12.2 Timer2 中断

Timer2 也可以产生可选的器件中断。Timer2 输出信号（TMR2 和 PR2 匹配时）为 4 位输出计数器 / 预分频器提供输入。此计数器产生的 TMR2 匹配中断标志位为 TMR2IF （PIR1<1>）。可以通过将 TMR2 匹配中断使能位 TMR2IE （PIE1<1>）置 1 来使能此中断。

可以通过后分频器控制位 T2OUTPS3:T2OUTPS0 （T2CON<6:3>）在 16 个后分频比值选项 （从 1:1 到 1:16）中选择其一。

12.3 TMR2 输出

TMR2 的不经分频的输出主要用于 CCP 模块，它用作 CCP 模块在 PWM 模式下工作时的时基。

还可选择将 Timer2 用作 MSSP 模块在 SPI 模式下的移位时钟源。**第 15.0 节 "主控同步串口（MSSP）模块"** 中提供了更多信息。

图 12-1： TIMER2 框图

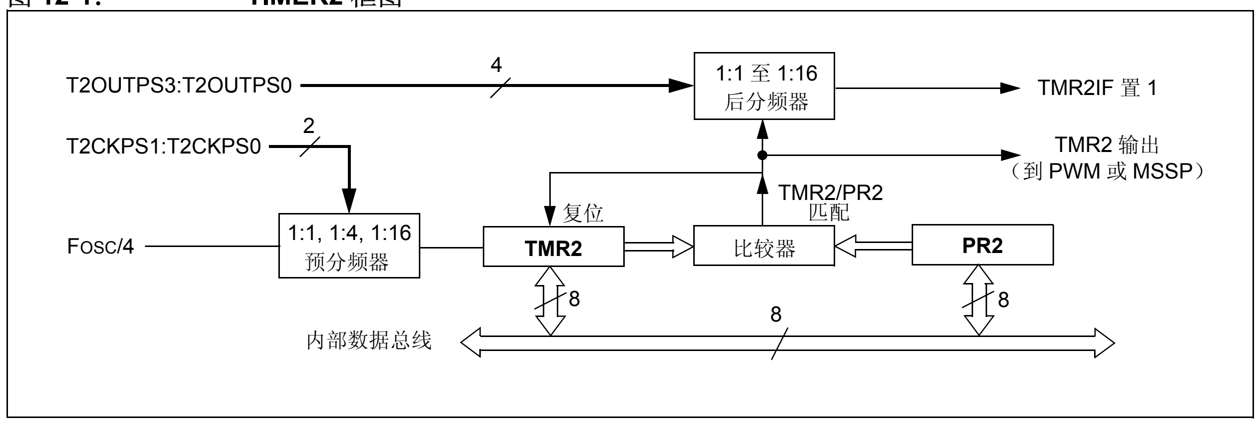


表 12-1： TIMER2 作为定时器 / 计数器时相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
PIR1	—	ADIF	RC1IF	TX1IF	SSPIF	CCP1IF	TMR2IF	TMR1IF	61
PIE1	—	ADIE	RC1IE	TX1IE	SSPIE	CCP1IE	TMR2IE	TMR1IE	61
IPR1	—	ADIP	RC1IP	TX1IP	SSPIP	CCP1IP	TMR2IP	TMR1IP	61
TMR2	Timer2 模块寄存器								60
T2CON	—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0	60
PR2	Timer2 周期寄存器								60

图注： — = 未用位，读为 0。在 Timer2 模块中未使用阴影单元。

 初稿

13.0 TIMER3 模块

Timer3 定时器 / 计数器模块具有以下特征：

- 可通过软件选择，作为 16 位定时器或计数器
- 可读写的 8 位寄存器 （TMR3H 和 TMR3L）
- 可选择使用器件时钟或 Timer1 内部振荡器作为内部或外部时钟源
- 溢出时中断
- CCP 特殊事件触发模块复位

图 13-1 所示为 Timer3 模块的简化框图。图 13-2 所示为此模块在读写模式下的工作原理框图。

Timer3 模块由 T3CON 寄存器 （寄存器 13-1）控制。该控制寄存器还可为 CCP 模块选择时钟源（欲知更多信息，请参见**第 14.1.1节“CCP模块和定时器资源”**）。

寄存器 13-1： T3CON：TIMER3 控制寄存器

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
RD16	T3CCP2	T3CKPS1	T3CKPS0	T3CCP1	$\overline{\text{T3SYNC}}$	TMR3CS	TMR3ON
bit 7							bit 0

bit 7 **RD16**：16 位读 / 写模式使能位
1 = 使能 Timer3 通过一次 16 位操作进行寄存器读 / 写
0 = 使能 Timer3 通过两次 8 位操作进行寄存器读 / 写

bit6, 3 **T3CCP2:T3CCP1**：CCPx 的时钟源 （是 Timer3 还是 Timer1）使能位
1x = Timer3 是 CCP 模块比较 / 捕捉的时钟源
01 = Timer3 是 CCP2 模块比较 / 捕捉的时钟源
Timer1 是 CCP1 模块比较 / 捕捉的时钟源
00 = Timer1 是 CCP 模块比较 / 捕捉的时钟源

bit5-4 **T3CKPS1:T3CKPS0**：Timer3 输入时钟预分频值选择位
11 = 1:8 预分频值
10 = 1:4 预分频值
01 = 1:2 预分频值
00 = 1:1 预分频值

bit2 **$\overline{\text{T3SYNC}}$**：Timer3 外部时钟输入同步控制位
（不适用于器件时钟来自 Timer1/Timer3 的场合。）
当 TMR3CS = 1 时：
1 = 不同步外部时钟输入
0 = 同步外部时钟输入
当 TMR3CS = 0 时：
忽略此位。当 TMR3CS = 0 时， Timer3 使用内部时钟。

bit1 **TMR3CS**：Timer3 时钟源选择位
1 = 使用 Timer1 振荡器或 T13CKI 引脚信号作为外部时钟输入（在第一个下降沿之后的上升沿开始计数）
0 = 内部时钟 （FOSC/4）

bit0 **TMR3ON**：Timer3 使能位
1 = 使能 Timer3
0 = 停止 Timer3

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

13.1 Timer3 工作原理

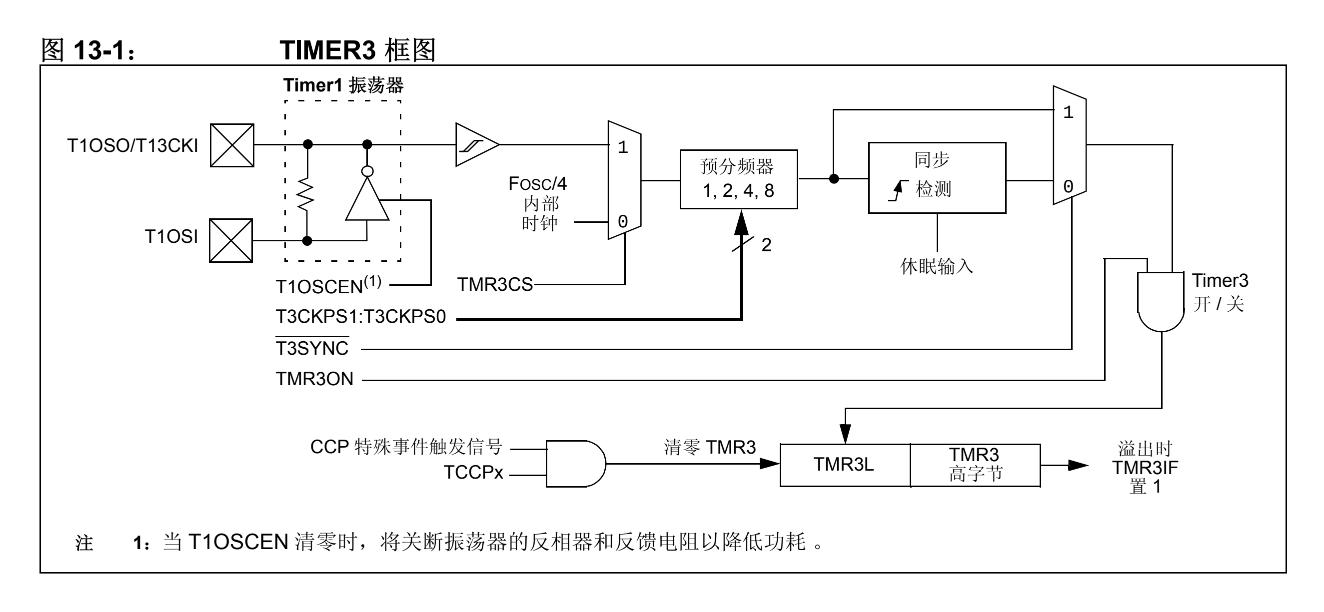
Timer3 有以下三种工作模式：

- 定时器
- 同步计数器
- 异步计数器

工作模式由时钟选择位 TMR3CS（T3CON<1>）决定。当 TMR3CS 清零（= 0）时，Timer3 在每个内部指令周期（Fosc/4）递增。当 TMR3CS 位置 1 时，Timer3 在 Timer1 外部时钟输入信号或 Timer1 振荡器输出信号（如果使能）的每个上升沿递增。

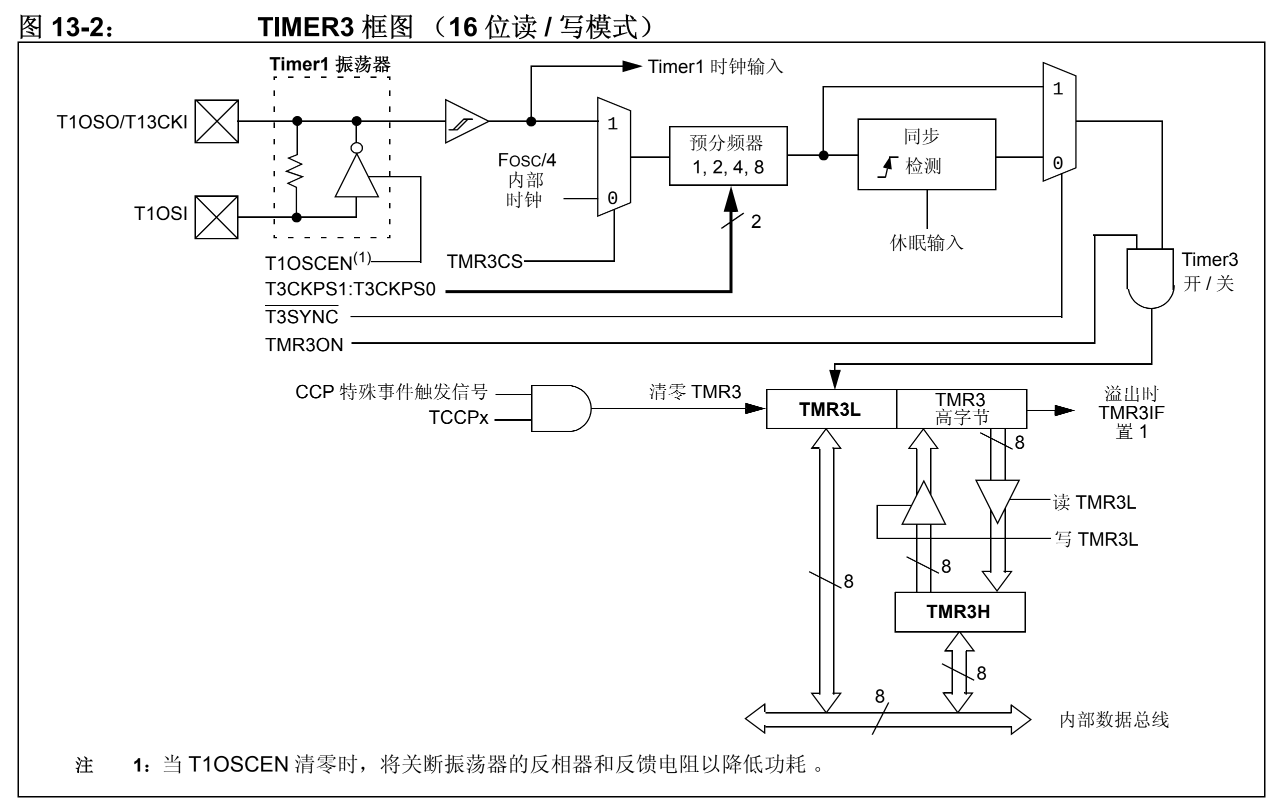
当使能 Timer1 时，RC1/T1OSI 和 RC0/T1OSO/T13CKI 引脚变为输入引脚。这意味着 TRISC<1:0> 的值被忽略并且这些引脚将读为 0。

图 13-1： TIMER3 框图



注 1：当 T1OSCEN 清零时，将关断振荡器的反相器和反馈电阻以降低功耗 。

图 13-2： TIMER3 框图 （16 位读 / 写模式）



注 1：当 T1OSCEN 清零时，将关断振荡器的反相器和反馈电阻以降低功耗 。

 初稿

13.2 Timer3 16 位读 / 写模式

可将 Timer3 配置为 16 位读写模式（见图 13-2）。当 RD16 控制位（T3CON<7>）置 1 时， TMR3H 的地址被映射到 Timer3 的高字节缓冲寄存器。对 TMR3L 的读操作将把 Timer3 的高字节装入 Timer3 高字节缓冲寄存器。这种方式使用户可以精确地读取 Timer3 的全部 16 位，而不需要像先读高字节再读低字节那样由于两次读取之间可能存在进位，而不得不验证读取的有效性。

对 Timer3 的高字节进行写操作也必须通过 TMR3H 缓冲寄存器进行。在写入 TMR3L 的同时，使用 TMR3H 的内容更新 Timer3 的高字节。这样允许用户将所有的 16 位值一次写入 Timer1 的高字节和低字节。

在该模式下不能直接读写 Timer3 的高字节，所有读写都必须通过 Timer3 高字节缓冲寄存器来进行。

写入 TMR3H 不会清零 Timer3 预分频器，只有在写 TMR3L 时才会清零该预分频器。

13.3 使用 Timer1 振荡器作为 Timer3 的时钟源

Timer1 内部振荡器可用作 Timer3 的时钟源。通过将 T1OSCEN（T1CON<3>）位置 1，可使能 Timer1 振荡器。要将它用作 Timer3 的时钟源还必须将 TMR3CS 位置 1。如前文所述，这样做也会将 Timer3 配置为在振荡器源的每个上升沿递增。

对 Timer1 振荡器的描述，请参见**第 11.0 节 “Timer1 模块”**。

13.4 Timer3 中断

TMR3 寄存器对（TMR3H:TMR3L）从 0000h 开始加 1，一直加到 FFFFh，然后溢出返回 0000h。如果使能了 Timer3 中断，该中断就会在溢出时产生，由中断标志位 TMR3IF（PIR2<0>）表示。可以通过对 Timer3 中断使能位 TMR3IE （PIE2<0>）置 1 或清零来使能或禁止该中断。

13.5 使用 CCP 特殊事件触发信号复位 Timer3

如果 CCP 模块被配置为产生特殊事件触发信号（CCP1M3:CCP1M0 或 CCP2M3:CCP2M0 = `1011`）的比较模式，该触发信号将复位 Timer3。如果使能了 A/D 模块，来自 CCP2 的触发信号还将启动 A/D 转换（欲知更多信息，请参见**第 14.3.4 节 “特殊事件触发器”**。）。

要使用这一功能，必须将 Timer3 配置为定时器或同步计数器。在这种情况下， CCPR2H:CCPR2L 这对寄存器实际上变成了 Timer3 的周期寄存器。

如果 Timer3 在异步计数器模式下运行，复位操作可能不起作用。

如果 Timer3 的写操作和特殊事件触发同时发生，则写操作优先。

> **注：** CCP2 模块的特殊事件触发信号不会将中断标志位 TMR3IF （PIR2<1>）置 1。

表 13-1： TIMER3 作为定时器 / 计数器时相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
PIR2	OSCFIF	CMIF	—	—	BCLIF	HLVDIF	TMR3IF	CCP2IF	61
PIE2	OSCFIE	CMIE	—	—	BCLIE	HLVDIE	TMR3IE	CCP2IE	61
IPR2	OSCFIP	CMIP	—	—	BCLIP	HLVDIP	TMR3IP	CCP2IP	61
TMR3L	16 位 TMR3 寄存器的低 8 位保持寄存器								61
TMR3H	16 位 TMR3 寄存器的高 8 位保持寄存器								61
T1CON	RD16	T1RUN	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{\text{T1SYNC}}$	TMR1CS	TMR1ON	60
T3CON	RD16	T3CCP2	T3CKPS1	T3CKPS0	T3CCP1	$\overline{\text{T3SYNC}}$	TMR3CS	TMR3ON	61

图注： — = 未用位，读为 0。在 Timer3 模块中未使用阴影单元。

注：

14.0 捕捉 / 比较 /PWM （CCP）模块

PIC18F6390/6490/8390/8490 器件具有两个 CCP （捕捉 / 比较 /PWM）模块，分别命名为 CCP1 和 CCP2。两个模块均可实现标准的捕捉、比较和脉宽调制（PWM）模式。

每个 CCP 模块包含一个 16 位寄存器，可用作 16 位捕捉寄存器、16 位比较寄存器或 PWM 主 / 从占空比寄存器。为简单起见，以下部分中所有对模块操作的描述均针对 CCP2，但同样适用于 CCP1。

寄存器 14-1： CCPxCON 寄存器 （CCP1 模块和 CCP2 模块）

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	DCxB1	DCxB0	CCPxM3	CCPxM2	CCPxM1	CCPxM0
bit 7							bit 0

bit7-6 未用位：读为 0

bit5-4 **DCxB1:DCxB0**：CCPx 模块的 PWM 占空比 bit 1 和 bit 0。
捕捉模式：
未用。
比较模式：
未用。
PWM 模式：
这两位是 10 位 PWM 占空比的两个最低位 （bit 1 和 bit 0）。占空比的高 8 位（DCx9:DCx2）在 CCPRxL 寄存器中。

bit3-0 **CCPxM3:CCPxM0**：CCPx 模块模式选择位
0000 = 禁用捕捉 / 比较 /PWM （复位 CCPx 模块）
0001 = 保留
0010 = 比较模式，匹配时输出电平翻转 （CCPxIF 位置 1）
0011 = 保留
0100 = 捕捉模式，每个下降沿捕捉
0101 = 捕捉模式，每个上升沿捕捉
0110 = 捕捉模式，每 4 个上升沿捕捉
0111 = 捕捉模式，每 16 个上升沿捕捉
1000 = 比较模式：初始化 CCP 引脚为低电平，比较匹配时强制 CCP 引脚为高电平（CCPIF 位置 1）
1001 = 比较模式：初始化 CCP 引脚为高电平，比较匹配时强制 CCP 引脚为低电平（CCPIF 位置 1）
1010 = 比较模式：比较匹配时产生软件中断 （CCPIF 位置 1， CCP 引脚反映 I/O 状态）
1011 = 比较模式：当 CCP2 发生匹配时触发特殊事件，复位定时器或启动 A/D 转换（CCPIF 位置 1）[1]
11xx = PWM 模式

注 1：CCP1M3:CCP1M0 = 1011在CCP1发生匹配事件时将只复位定时器而不启动A/D转换。

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

14.1 CCP 模块配置

每个捕捉 / 比较 /PWM 模块均与一个控制寄存器（通常为 CCPxCON）和一个数据寄存器（CCPRx）相对应。数据寄存器由两个 8 位寄存器组成：CCPRxL （低字节）和 CCPRxH（高字节）。所有寄存器都是可读写的。

14.1.1 CCP 模块和定时器资源

CCP 模块根据选定的模式使用 Timer1、 Timer2 或 Timer3。该模块在捕捉或比较模式下使用 Timer1 和 Timer3 而在 PWM 模式下使用 Timer2。

表 14-1： CCP 模块—定时器资源

CCP 模式	定时器资源
捕捉	Timer1 或 Timer3
比较	Timer1 或 Timer3
PWM	Timer2

要将哪个特定的定时器分配给 CCP 模块由 T3CON 寄存器（寄存器 13-1）中的“Timer-to-CCP”使能位决定。若将两个 CCP 模块配置为同时工作在相同的模式 （捕捉 / 比较或 PWM）下，那么这两个模块可共享相同的定时器资源。表 14-2 总结了这两个模块间的相互关系。

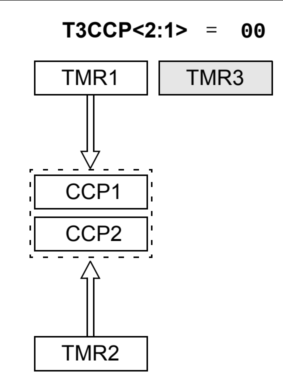
根据选定的配置，并且模块配置相同的情况下 （捕捉 / 比较或 PWM）最多可有 4 个定时器同时工作。图 14-1 所示为可能的配置。

14.1.2 CCP2 引脚分配

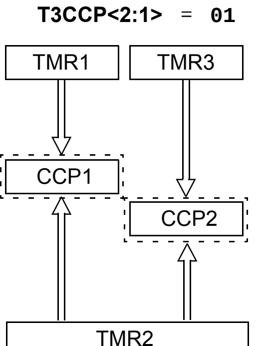
可根据器件配置改变 CCP2 （捕捉输入、比较和 PWM 输出）的引脚分配。 CCP2MX 配置位决定哪个引脚与 CCP2 复用。默认情况下，该引脚分配给 RC1 （CCP2MX = 1）。如果清零该配置位，CCP2 将与 RE7 引脚复用。

改变 CCP2 的引脚分配并不会自动改变对端口引脚的配置。用户必须始终确认与 CCP2 操作相对应的 TRIS 寄存器配置正确。

图 14-1： CCP 与定时器的互连配置

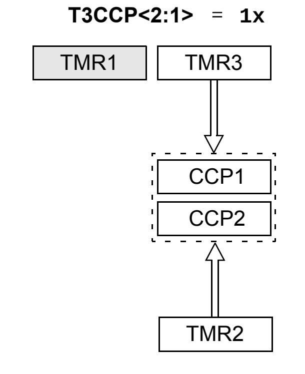


Timer1 用于所有 CCP 模块的捕捉和比较操作。Timer2 用于所有 CCP 模块的 PWM 操作。模块可能会共享任一定时器作为通用时基。



Timer1 用于 CCP1 的捕捉和比较操作而 Timer3 用于 CCP2。

如果两个模块均处于 PWM 模式，那么它们将使用 Timer2 作为通用时基。



Timer3 用于所有 CCP 模块的捕捉和比较操作。Timer2 用于所有 CCP 模块的 PWM 操作。模块可能会共享任一定时器作为通用时基。

表 **14-2**： **CCP1** 和 **CCP2** 在使用定时器资源方面的相互关系

CCP1 模式	**CCP2 模式**	相互关系
捕捉	捕捉	每个模块都可用 TMR1 或 TMR3 作为时基。每个 CCP 的时基也可以各不相同。
捕捉	比较	可将 CCP2 配置为特殊事件触发器用以复位 TMR1 或 TMR3 （取决于所使用的时基），也可用于自动触发 A/D 转换。若 CCP1 使用与 CCP2 相同的定时器作为时基，上述操作可能会对 CCP1 产生影响。
比较	捕捉	可将 CCP1 配置为特殊事件触发器用以复位 TMR1 或 TMR3 （取决于所使用的时基）。如果 CCP2 使用与 CCP1 相同的定时器作为时基，上述操作可能会对 CCP2 产生影响。
比较	比较	每个模块均可配置为特殊事件触发器用以复位时基。 CCP2 还可自动触发 A/D 转换。若两个模块使用相同的时基，可能会发生冲突。
捕捉	PWM*	无
比较	PWM*	无
PWM*	捕捉	无
PWM*	比较	无
PWM*	PWM	两个 PWM 具有相同的频率和更新速率 （TMR2 中断）。

* 包括标准和增强型 PWM。

14.2 捕捉模式

在捕捉模式下，当在 CCP2 引脚 （RC1 或 RE7 引脚，取决于器件配置）上发生事件时，CCPR2H:CCPR2L 寄存器对捕捉 TMR1 或 TMR3 的 16 位值。事件定义为下列情况之一：

- 每个下降沿
- 每个上升沿
- 每 4 个上升沿
- 每 16 个上升沿

事件类型由模式选择位 CCP2M3:CCP2M0 （CCP2CON<3:0>）选择。当完成一次捕捉时，中断请求标志位 CCP2IF （PIR2<1>）置 1，它必须用软件清零。如果在读取寄存器 CCPR2 值之前发生了另一个捕捉，那么之前捕捉的值将会被覆盖。

14.2.1 CCP 引脚配置

在捕捉模式下，应通过将相应的 TRIS 方向位置 1 把 CCPx 引脚配置为输入。

> **注：** 若将 RC1/CCP2 或 RE7/CCP2 引脚配置为输出，对该端口的写操作将会产生捕捉条件。

14.2.2 TIMER1/TIMER3 模式选择

用于捕捉功能的定时器（Timer1 和 / 或 Timer3）必须运行在定时器模式或同步计数器模式下。在异步计数器模式下， CCP 模块无法进行捕捉操作。可在 T3CON 寄存器中选择用于 CCP 模块的定时器。（见第 **14.1.1** 节 **"CCP 模块和定时器资源"**）。

14.2.3 软件中断

当捕捉模式改变时，可能会产生错误的捕捉中断。用户应该保持 CCP2IE 位 （PIE2<1>）清零以避免错误中断，并且还应该在工作模式改变后清零标志位 CCP2IF。

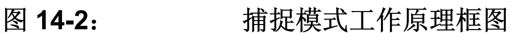
14.2.4 CCP 预分频器

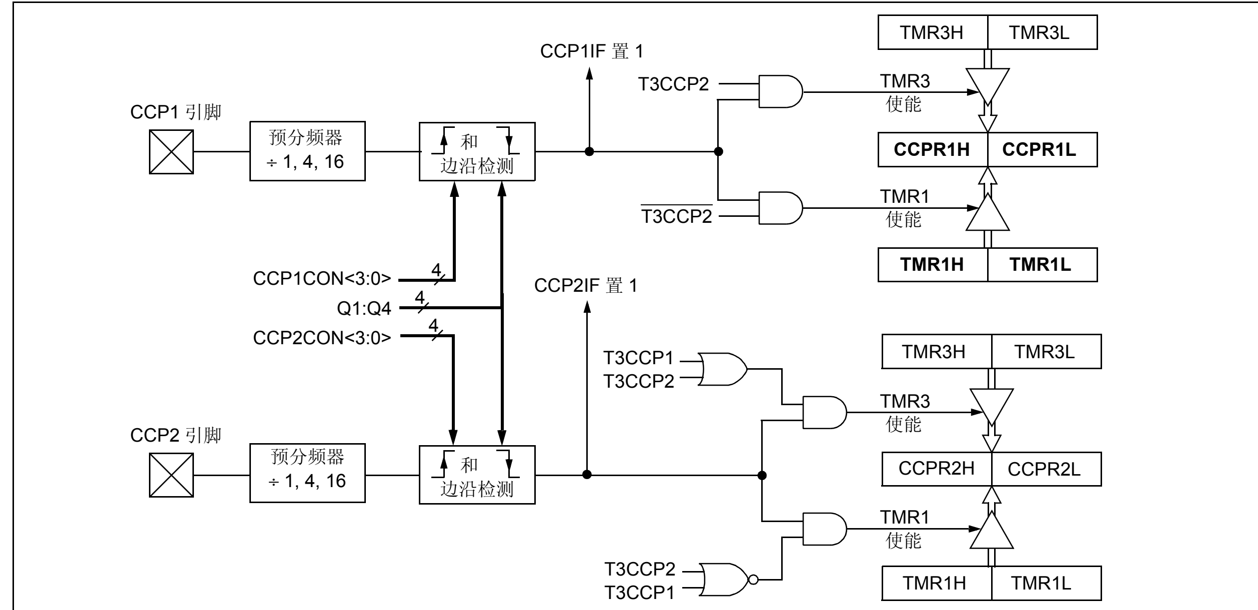
在捕捉模式下有四种预分频比值设置，它们可作为工作模式的一部分由模式选择位 （CCP2M3:CCP2M0）选择。只要关闭 CCP 模块或者 CCP 模块不处于捕捉模式，预分频计数器就将被清零。这意味着任何复位都将清零预分频计数器。

在两个预分频器之间切换会产生中断，且不会清零预分频计数器。切换后的第一次捕捉可能来自于一个非零的预分频器。例 14-1 是切换捕捉预分频器时建议采用的方法。该例子使预分频计数器清零且不会产生错误中断。

例 14-1： 改变捕捉预分频比值

```
CLRF   CCP2CON      ; Turn CCP module off
MOVLW  NEW_CAPT_PS  ; Load WREG with the
                    ; new prescaler mode
                    ; value and CCP ON
MOVWF  CCP2CON      ; Load CCP2CON with
                    ; this value
```


图 14-2： 捕捉模式工作原理框图



14.3 比较模式

在比较模式下， 16 位 CCPR2 寄存器的值不断与一对 TMR1 或 TMR3 寄存器的值比较。当两者匹配时，CCPx 引脚将被：

- 驱动为高电平
- 驱动为低电平
- 电平翻转 （高电平变为低电平或低电平变为高电平）
- 保持不变 （即呈现 I/O 锁存器状态）

引脚动作取决于模式选择位 （CCP2M3:CCP2M0）的值。同时，中断标志位 CCP2IF 置 1。

14.3.1 CCP 引脚配置

在比较模式下，用户必须通过将相应的 TRIS 位清零，把 CCPx 引脚配置为输出引脚。

注：	清零 CCP2CON 寄存器将把 RC1 或 RE7 比较输出锁存器 （取决于器件配置）强制设为默认的低电平状态。这不是 PORTC 或 PORTE I/O 数据锁存器。

14.3.2 TIMER1/TIMER3 模式选择

如果 CCP 模块使用比较功能，Timer1 和 / 或 Timer3 必须工作在定时器模式或同步计数器模式下。在异步计数器模式下， CCP 模块可能无法进行比较操作。

14.3.3 软件中断模式

当选择了生成软件中断模式时 （CCP2M3:CCP2M0 = `1010`）， CCP2 引脚上的电平不受影响。如果已经使能，将仅产生 CCP 中断并将 CCP2IE 置 1。

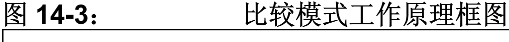
14.3.4 特殊事件触发器

两个 CCP 模块均配备了一个特殊事件触发器。在比较模式下可产生内部硬件信号以触发其他模块动作。通过选择比较特殊事件触发模式 （CCP2M3:CCP2M0 = `1011`），使能特殊事件触发器。

对于任何一个 CCP 模块，无论当前使用哪个定时器资源作为模块的时基，特殊事件触发器将把对应的定时寄存器复位。这样 CCPRx 寄存器可用作两个定时器的可编程周期寄存器。

CCP2 特殊事件触发器还能启动 A/D 转换。要实现此功能，必须首先使能 A/D 转换器。

注：	CCP1 特殊事件触发器只复位 Timer1/Timer3，即使在使能了 A/D 转换器也不能启动 A/D 转换。


图 14-3： 比较模式工作原理框图

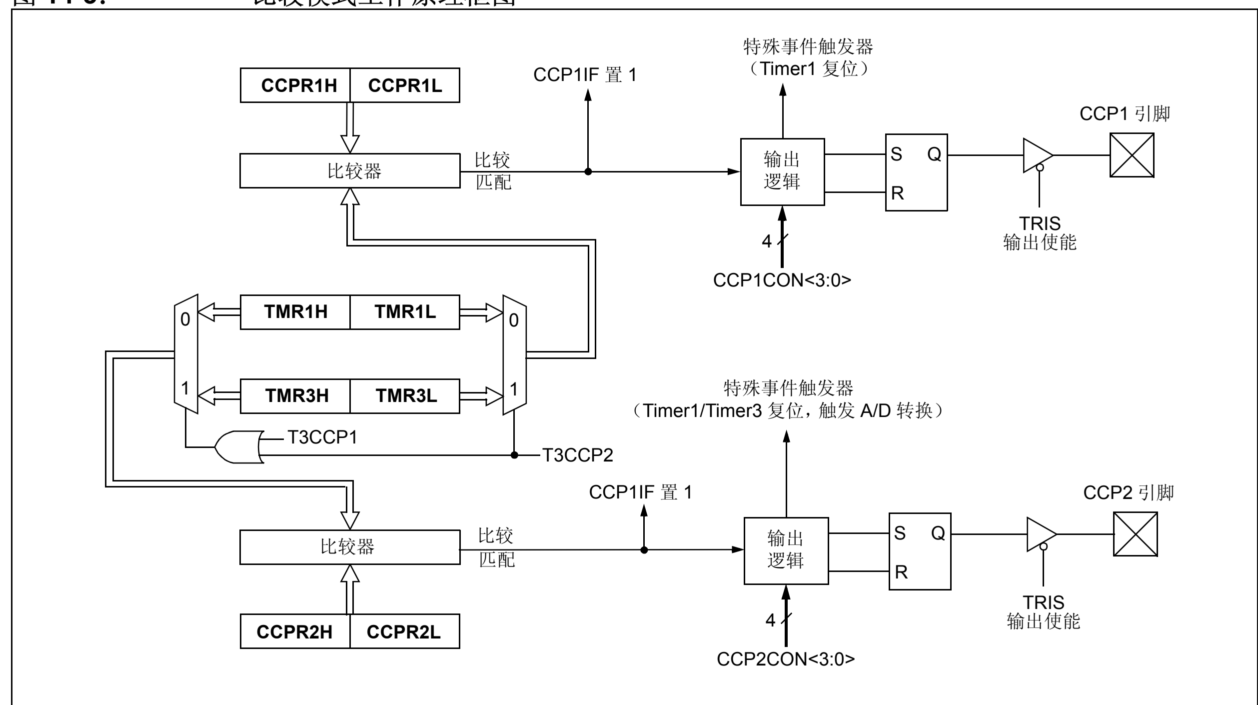


表 **14-3：** 与捕捉、比较、 **TIMER1** 和 **TIMER3** 相关的寄存器

名称	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	复位值所在的页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
RCON	IPEN	SBOREN	—	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$	60
PIR1	—	ADIF	RC1IF	TX1IF	SSPIF	CCP1IF	TMR2IF	TMR1IF	61
PIE1	—	ADIE	RC1IE	TX1IE	SSPIE	CCP1IE	TMR2IE	TMR1IE	61
IPR1	—	ADIP	RC1IP	TX1IP	SSPIP	CCP1IP	TMR2IP	TMR1IP	61
PIR2	OSCFIF	CMIF	—	—	BCLIF	HLVDIF	TMR3IF	CCP2IF	61
PIE2	OSCFIE	CMIE	—	—	BCLIE	HLVDIE	TMR3IE	CCP2IE	61
IPR2	OSCFIP	CMIP	—	—	BCLIP	HLVDIP	TMR3IP	CCP2IP	61
TRISC	PORTC 数据方向寄存器								62
TRISE	PORTE 数据方向寄存器				—	—	—	—	62
TMR1L	16 位 TMR1 寄存器的低字节保持寄存器								60
TMR1H	16 位 TMR1 寄存器的高字节保持寄存器								60
T1CON	RD16	T1RUN	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{T1SYNC}$	TMR1CS	TMR1ON	60
TMR3H	Timer3 寄存器的高字节								61
TMR3L	Timer3 寄存器的低字节								61
T3CON	RD16	T3CCP2	T3CKPS1	T3CKPS0	T3CCP1	$\overline{T3SYNC}$	TMR3CS	TMR3ON	61
CCPR1L	捕捉 / 比较 /PWM 寄存器 1 （LSB）								61
CCPR1H	捕捉 / 比较 /PWM 寄存器 1 （MSB）								61
CCP1CON	—	—	DC1B1	DC1B0	CCP1M3	CCP1M2	CCP1M1	CCP1M0	61
CCPR2L	捕捉 / 比较 /PWM 寄存器 2 （LSB）								61
CCPR2H	捕捉 / 比较 /PWM 寄存器 2 （MSB）								61
CCP2CON	—	—	DC2B1	DC2B0	CCP2M3	CCP2M2	CCP2M1	CCP2M0	61

图注： — = 未用位，读为 0。捕捉 / 比较、 Timer1 或 Timer3 不使用阴影单元。

注 1： 64 引脚器件不使用这些位，它们将始终保持清零。

14.4 PWM 模式

在脉宽调制（PWM）模式下， CCP2 引脚会产生最多 10 位分辨率的 PWM 输出信号。由于 CCP2 引脚与 PORTC 或 PORTE 数据锁存器复用，因此必须清零相应的 TRIS 位以使 CCP2 引脚为输出引脚。

注：	清零 CCP2CON 寄存器将把 RC1 或 RE7 输出锁存器（取决于器件配置）强制设为默认的低电平状态。这不是 PORTC 或 PORTE I/O 数据锁存器。

图 14-4 所示为 PWM 模式下 CCP 模块的简化框图。

如需了解如何配置 CCP2 模块使之工作于 PWM 模式的步骤，请参见第 **14.4.3 节 “设置 PWM 工作模式”**。

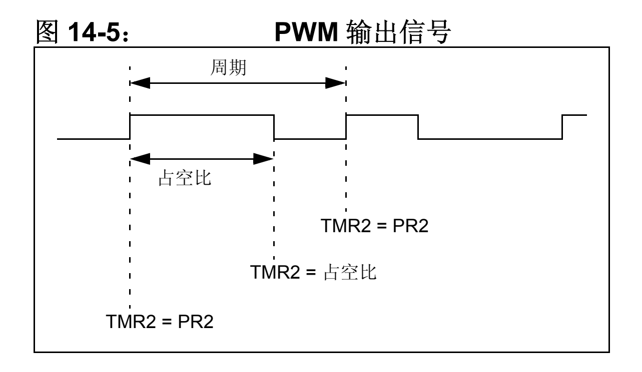
图 14-4： 简化的 PWM 框图

占空比寄存器
CCP1CON<5:4>
CCPR1L
CCPR1H（从）
比较器
R
Q
RC2/CCP1
TMR2
（注 1）
S
TRISC<2>
比较器
清零定时器、CCP1 引脚并锁存 D.C.
PR2

注 1：8 位 TMR2 值与内部 Q 时钟或预分频器中的 2 位一起构成 10 位时基。

PWM 输出信号（图 14-5）具有一个时基（周期）和一段输出保持高电平的时间（占空比）。 PWM 信号的频率是周期的倒数（1/ 周期）。

图 14-5： PWM 输出信号



14.4.1 PWM 周期

可通过写 PR2 寄存器指定 PWM 周期。用以下公式计算 PWM 周期：

公式 14-1：

PWM 周期 = [（PR2）+ 1] • 4 • TOSC •（TMR2 预分频值）

PWM 频率定义为 1 / [PWM 周期]。

当 TMR2 中的值与 PR2 中的值相等时，在下一个计数周期中将发生以下三个事件：

- TMR2 被清零
- CCP2 引脚置 1（例外：如果 PWM 占空比为 0%， CCP2 引脚将不会置 1）
- PWM 占空比将从 CCPR2L 锁存到 CCPR2H

注：	PWM 频率不是由 Timer2 后分频器决定的（见第 **12.0 节 “Timer2 模块”**）。 后分频器可用不同于 PWM 输出频率的频率进行数据更新。

14.4.2 PWM 占空比

通过写 CCPR2L 寄存器和 CCP2CON<5:4> 位来指定 PWM 占空比。分辨率最高可达 10 位。CCPR2L 包含八个 MSb 而 CCP2CON<5:4> 包含两个 LSb。这 10 位 PWM 占空比值表示为 CCPR2L:CCP2CON<5:4>。计算占空比的公式如下：

公式 14-2：

$$\text{PWM 占空比} = (\text{CCPR2L:CCP2CON<5:4>}) \cdot T_{OSC} \cdot (\text{TMR2 预分频值})$$

可以在任何时候写入 CCPR2L 和 CCP2CON<5:4>，但是在 PR2 和 TMR2 发生匹配（即周期结束）前占空比值不会被锁存到 CCPR2H 中。在 PWM 模式下，CCPR2H 是只读寄存器。

CCPR2H 寄存器和一个 2 位的内部锁存器用于给 PWM 占空比提供双重缓冲。这种双重缓冲结构非常重要，它可以避免在 PWM 工作过程中产生毛刺。

当 CCPR2H 和 2 位锁存器的值与 TMR2（连有内部 2 位 Q 时钟或 TMR2 预分频器中的 2 位）匹配时，CCP2 引脚被清零。

对于给定的 PWM 频率，最大的 PWM 分辨率（位）由以下公式给出：

公式 14-3：

$$\text{PWM 分辨率（最大）} = \frac{\log\left(\frac{F_{OSC}}{F_{PWM}}\right)}{\log(2)} \text{ 位}$$

> **注：** 如果 PWM 占空比的值大于 PWM 周期，CCP2 引脚将不会被清零。

表 14-4： 40 MHz 下的 PWM 频率和分辨率示例

PWM 频率	2.44 kHz	9.77 kHz	39.06 kHz	156.25 kHz	312.50 kHz	416.67 kHz
定时器预分频值（1,4,16）	16	4	1	1	1	1
PR2 值	FFh	FFh	FFh	3Fh	1Fh	17h
最高分辨率（位）	14	12	10	8	7	6.58

14.4.3 设置 PWM 工作模式

当配置 CCP2 模块使之工作于 PWM 模式时应遵循以下步骤：

1. 通过写 PR2 寄存器设置 PWM 周期。
2. 通过写 CCPR2L 寄存器和 CCP2CON<5:4> 位来指定 PWM 占空比。
3. 通过清零相应的 TRIS 位将 CCP2 引脚设为输出引脚。
4. 通过写 T2CON 设置 TMR2 预分频值并随后使能 Timer2。
5. 配置 CCP2 模块使之工作于 PWM 模式。

表 14-5： 与 PWM 和 TIMER2 相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在的页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
RCON	IPEN	SBOREN	—	$\overline{\text{RI}}$	$\overline{\text{TO}}$	$\overline{\text{PD}}$	$\overline{\text{POR}}$	$\overline{\text{BOR}}$	60
PIR1	—	ADIF	RC1IF	TX1IF	SSPIF	CCP1IF	TMR2IF	TMR1IF	61
PIE1	—	ADIE	RC1IE	TX1IE	SSPIE	CCP1IE	TMR2IE	TMR1IE	61
IPR1	—	ADIP	RC1IP	TX1IP	SSPIP	CCP1IP	TMR2IP	TMR1IP	61
TRISC	PORTC 数据方向寄存器								62
TRISE	PORTE 数据方向寄存器				—	—	—	—	62
TMR2	Timer2 模块寄存器								60
PR2	Timer2 模块周期寄存器								60
T2CON	—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0	60
CCPR1L	捕捉 / 比较 /PWM 寄存器 1 （LSB）								61
CCPR1H	捕捉 / 比较 /PWM 寄存器 1 （MSB）								61
CCP1CON	—	—	DC1B1	DC1B0	CCP1M3	CCP1M2	CCP1M1	CCP1M0	61
CCPR2L	捕捉 / 比较 /PWM 寄存器 2 （LSB）								61
CCPR2H	捕捉 / 比较 /PWM 寄存器 2 （MSB）								61
CCP2CON	—	—	DC2B1	DC2B0	CCP2M3	CCP2M2	CCP2M1	CCP2M0	61

图注： — = 未用位 , 读为 0。 PWM 或 Timer2 不使用阴影单元。

注：

15.0 主控同步串口 （MSSP）模块

15.1 主控 SSP （MSSP）模块概述

主控同步串口（Master Synchronous Serial Port，MSSP）模块是用于同其他外设或单片机器件进行通信的串行接口。这些外设器件可以是串行 EEPROM、移位寄存器、显示驱动器和 A/D 转换器等等。MSSP 模块有下列两种工作模式：

- 串行外设接口 （SPI）
- I²C 模块
  - 全主控模式
  - 从动模式 （支持广播地址呼叫）

I²C 接口硬件上支持下列模式：

- 主控模式
- 多主器件模式
- 从动模式

15.2 控制寄存器

MSSP 模块有三个相关的寄存器，包括一个状态寄存器（SSPSTAT）和两个控制寄存器（SSPCON1 和 SSPCON2）。根据 MSSP 模块是在 SPI 模式还是 I²C 模式下工作，这些寄存器的用途及它们各自的配置位将完全不同。

下面各节会提供更多详细信息。

15.3 SPI 模式

SPI 模式允许同时同步发送和接收 8 位数据。器件支持 SPI 的所有四种模式。通常使用以下三个引脚来实现通信：

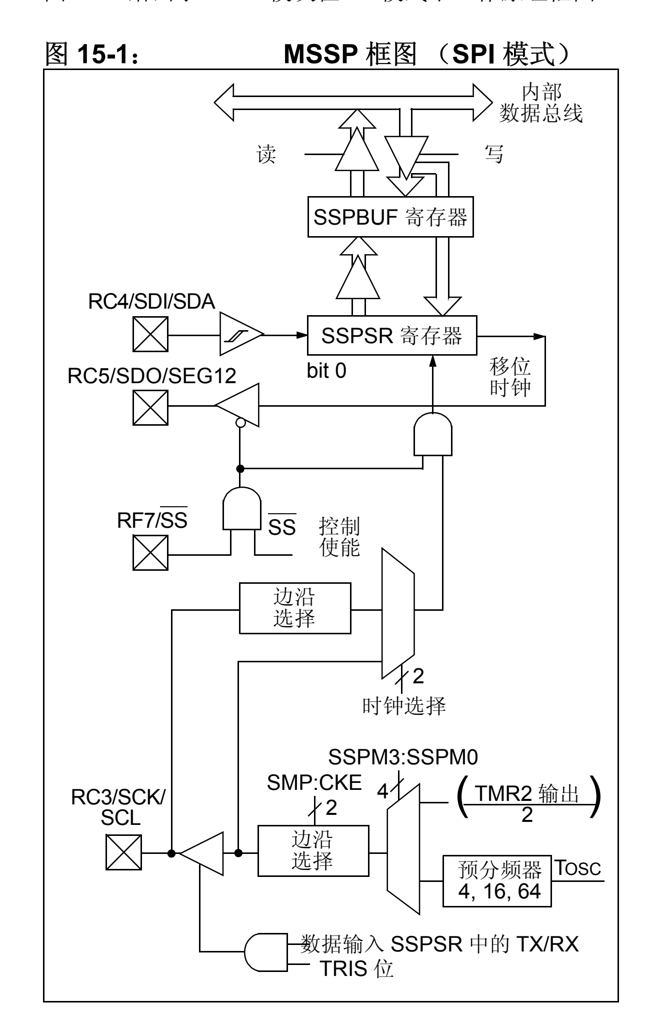
- 串行数据输出 （Serial Data Out， SDO）－ RC5/SDO/SEG12
- 串行数据输入 （Serial Data In， SDI）－ RC4/SDI/SDA
- 串行时钟 （Serial Clock， SCK）－ RC3/SCK/SCL

此外，当处于从动工作模式时要使用第 4 根引脚：

- 从动选择 （$\overline{SS}$）－ RF7/$\overline{SS}$

图 15-1给出了MSSP模块在SPI模式下工作原理框图。

图 15-1： MSSP 框图 （SPI 模式）



15.3.1 寄存器

MSSP 模块有四个寄存器用于 SPI 工作模式。这些寄存器包括：

- MSSP 控制寄存器 1 （SSPCON1）
- MSSP 状态寄存器 （SSPSTAT）
- 串行接收 / 发送缓冲寄存器 （SSPBUF）
- MSSP 移位寄存器 （SSPSR）－不可直接访问

SSPCON1 和 SSPSTAT 是 SPI 模式下的控制寄存器和状态寄存器。SSPCON1 寄存器是可读写的。SSPSTAT 的低 6 位是只读的，而高 2 位是可读写的。

SSPSR 是用来将数据移入或移出的移位寄存器。SSPBUF 是缓冲寄存器，可用于数据字节的写入或读出。

接收数据时，SSPSR 和 SSPBUF 共同构成一个双重缓冲接收器。当 SSPSR 接收到一个完整的字节之后，该字节会被送入SSPBUF，同时将中断标志位SSPIF置1。

在数据发送过程中， SSPBUF 不是双重缓冲的，对 SSPBUF 的写操作将同时写入 SSPBUF 和 SSPSR。

寄存器 15-1： SSPSTAT：MSSP 状态寄存器 （SPI 模式）

R/W-0	R/W-0	R-0	R-0	R-0	R-0	R-0	R-0
SMP	CKE	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF

bit 7 … bit 0

bit 7 **SMP**：采样位
SPI 主控模式：
1 = 在数据输出时间的末端采样输入数据
0 = 在数据输出时间的中间采样输入数据
SPI 从动模式：
当 SPI 工作在从动模式时，必须将 SMP 清零。

bit 6 **CKE**：SPI 时钟边沿选择位
当 CKP = 0 时：
1 = 在 SCK 的上升沿发送数据
0 = 在 SCK 的下降沿发送数据
当 CKP = 1 时：
1 = 在 SCK 的下降沿发送数据
0 = 在 SCK 的上升沿发送数据

bit 5 **D/$\overline{A}$**：数据 / $\overline{地址}$位
只在 I^2C 模式下使用。

bit 4 **P**：停止位
只在 I^2C 模式下使用。当禁止 MSSP 模块 （SSPEN 清零）时，该位被清零。

bit 3 **S**：启动位
只在 I^2C 模式下使用。

bit 2 **R/$\overline{W}$**：读 / $\overline{写}$位信息
只在 I^2C 模式下使用。

bit 1 **UA**：更新地址位
只在 I^2C 模式下使用。

bit 0 **BF**：缓冲器满状态位 （仅用于接收模式）
1 = 接收完成， SSPBUF 满
0 = 接收未完成， SSPBUF 空

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

寄存器 15-2： SSPCON1：MSSP 控制寄存器 1 （SPI 模式）

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0
bit 7							bit 0

bit 7 **WCOL**：写冲突检测位 （仅用于发送模式下）
1 = 正在发送前一个字时，又有数据写入 SSPBUF 寄存器
（必须用软件清零）
0 = 未发生冲突

bit 6 **SSPOV**：接收溢出指示位
SPI 从动模式：
1 = SSPBUF 中仍保存前一数据时，又接收到一个新的字节。如果溢出，SSPSR 中的数据会丢失。溢出只会在从动模式下发生。即使只是发送数据，用户也必须读 SSPBUF，以避免将溢出标志位置 1 （该位必须由软件清零）。
0 = 无溢出

注：在主控模式下，溢出位不会被置 1，因为每次接收和发送新数据都是通过写入 SSPBUF 寄存器启动的。

bit 5 **SSPEN**：同步串口使能位
1 = 使能串口并将 SCK、 SDO、 SDI 和 $\overline{SS}$ 配置为串口引脚
0 = 禁止串口并将上述引脚配置为 I/O 端口引脚

注：当使能时，必须将这些引脚正确地配置为输入或输出。

bit 4 **CKP**：时钟极性选择位
1 = 空闲状态 时，时钟为高电平
0 = 空闲状态时，时钟为低电平

bit 3-0 **SSPM3:SSPM0**：同步串口模式选择位
0101 = SPI 从动模式，时钟 = SCK 引脚，禁止 $\overline{SS}$ 引脚控制，可将 $\overline{SS}$ 用作 I/O 引脚
0100 = SPI 从动模式，时钟 = SCK 引脚，使能 $\overline{SS}$ 引脚控制
0011 = SPI 主控模式，时钟 = TMR2 输出 /2
0010 = SPI 主控模式，时钟 = Fosc/64
0001 = SPI 主控模式，时钟 = Fosc/16
0000 = SPI 主控模式，时钟 = Fosc/4

注： 在此未列出的位组合用于保留或仅在 I^2C 模式下使用。

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

15.3.2 工作原理

当初始化 SPI 时，通过对相应的控制位（SSPCON1<5:0> 和 SSPSTAT<7:6>）编程来设置选项。这些控制位用于设置以下选项：

- 主控模式 （SCK 作为时钟输出）
- 从动模式 （SCK 作为时钟输入）
- 时钟极性 （SCK 的空闲状态）
- 输入数据的采样相位 （数据输出时间的中间或末端）
- 时钟边沿 （在 SCK 的上升沿 / 下降沿输出数据）
- 时钟速率 （仅用于主控模式）
- 从动选择模式 （仅用于从动模式）

MSSP 模块由一个发送 / 接收移位寄存器（SSPSR）和一个缓冲寄存器 （SSPBUF）组成。 SSPSR 对进出器件的数据进行移位，最高有效位在前。在新数据接收完毕前， SSPBUF 保存上次写入 SSPSR 的数据。一旦 8 位数据接收完毕，该字节就被移入 SSPBUF 寄存器。然后，缓冲器满检测位 BF（SSPSTAT<0>) 和中断标志位 SSPIF 被置 1。这种双重缓冲数据接收方式（SSPBUF），允许在 CPU 读取刚接收的数据之前，就开始接收下一个字节。在数据发送 / 接收期间，任何试图写 SSPBUF 寄存器的操作都无效，并且写冲突检测位 WCOL （SSPCON1<7>）被置 1。用户必须用软件将 WCOL 位清零才能判断以后对 SSPBUF 寄存器的写入是否成功。

为确保应用软件能有效地接收数据，在下一个要发送的数据字节写入 SSPBUF 之前，读取 SSPBUF 中现有的数据。缓冲器满位 BF （SSPSTAT<0>）用于表示何时 SSPBUF 载入了接收到的数据 （发送完成） 。当 SSPBUF 中的数据被读取后，BF 位即被清零。如果 SPI 仅仅作为一个发送器，则不必理会该位。通常，可用 MSSP 中断来判断发送 / 接收是否已完成。必须读取和 / 或写入 SSPBUF。如果不打算使用中断，用软件查询的方法同样可确保不会发生写冲突。例 15-1 举例说明了装载 SSPBUF （SSPSR）进行数据发送的过程。

不能直接读写 SSPSR 寄存器，只能通过寻址 SSPBUF 寄存器来访问。此外，MSSP 状态寄存器（SSPSTAT）指示各种状态。

例 15-1：　　装载 SSPBUF （SSPSR）寄存器

```
LOOP    BTFSS   SSPSTAT, BF     ;Has data been received (transmit complete)?
        BRA     LOOP            ;No
        MOVF    SSPBUF, W       ;WREG reg = contents of SSPBUF
        MOVWF   RXDATA          ;Save in user RAM, if data is meaningful
        MOVF    TXDATA, W       ;W reg = contents of TXDATA
        MOVWF   SSPBUF          ;New data to xmit
```

15.3.3 使能 SPI I/O

要使能串口， SSP 使能位 SSPEN （SSPCON1<5>）必须置 1。要复位或重新配置 SPI 模式，要先将 SSPEN 位清零，重新初始化 SSPCON 寄存器，然后将 SSPEN 位置 1。这将把 SDI、SDO、SCK 和 $\overline{SS}$ 引脚配置为串口引脚。要让上述引脚充当串口，必须正确设置其中一些引脚的数据方向位 （在 TRIS 寄存器中）。

- SDI 由 SPI 模块自动控制
- SDO 必须将 TRISC<5> 位清零
- SCK （主控模式）必须将 TRISC<3> 位清零
- SCK （从动模式）必须将 TRISC<3> 位置 1
- $\overline{SS}$ 必须将 TRISF<7> 位置 1

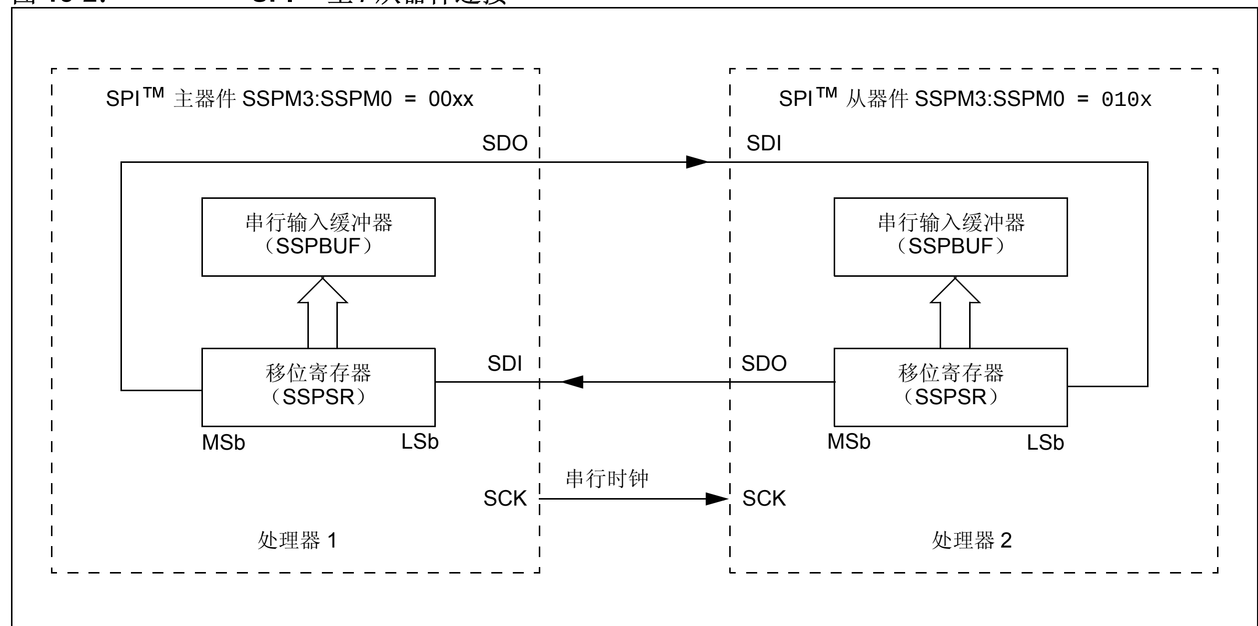
对于不需要的串口功能，可通过将对应的数据方向寄存器 （TRIS）设置为相反值来屏蔽。

15.3.4 典型连接

图 15-2 给出两个单片机之间的典型连接。主器件 （处理器 1）通过发送 SCK 信号来启动数据传输。在两个处理器的移位寄存器之间，数据在编程设定的时钟边沿被传送，并在相反的时钟边沿被锁存。必须将两个处理器的时钟极性（CKP）设置为相同，这样就可以同时收发数据。数据是否有效，取决于应用软件。这就导致以下三种数据传输情形：

- 主器件发送数据 — 从器件发送无效 （Dummy）数据
- 主器件发送数据 — 从器件发送数据
- 主器件发送无效数据 — 从器件发送数据


图 15-2： SPI™ 主 / 从器件连接



15.3.5 主控模式

因为由主器件控制 SCK 信号，所以它可以在任意时刻启动数据传输。主器件根据软件协议确定从器件（图 15-2 中的处理器 2）应在何时广播数据。

在主控模式下，数据一旦写入SSPBUF寄存器就开始发送或接收。如果只打算将 SPI 作为接收器，则可以禁止 SDO 输出（将其编程设置为输入）。SSPSR 寄存器按设置的时钟速率，对 SDI 引脚上的信号进行连续移位输入。每收到一个字节，就将其装入 SSPBUF 寄存器，就像接收到普通字节一样 （中断和状态位相应置 1）。这在以 "线路活动监控"（Line Activity Monitor）方式工作的接收器应用中很有用。

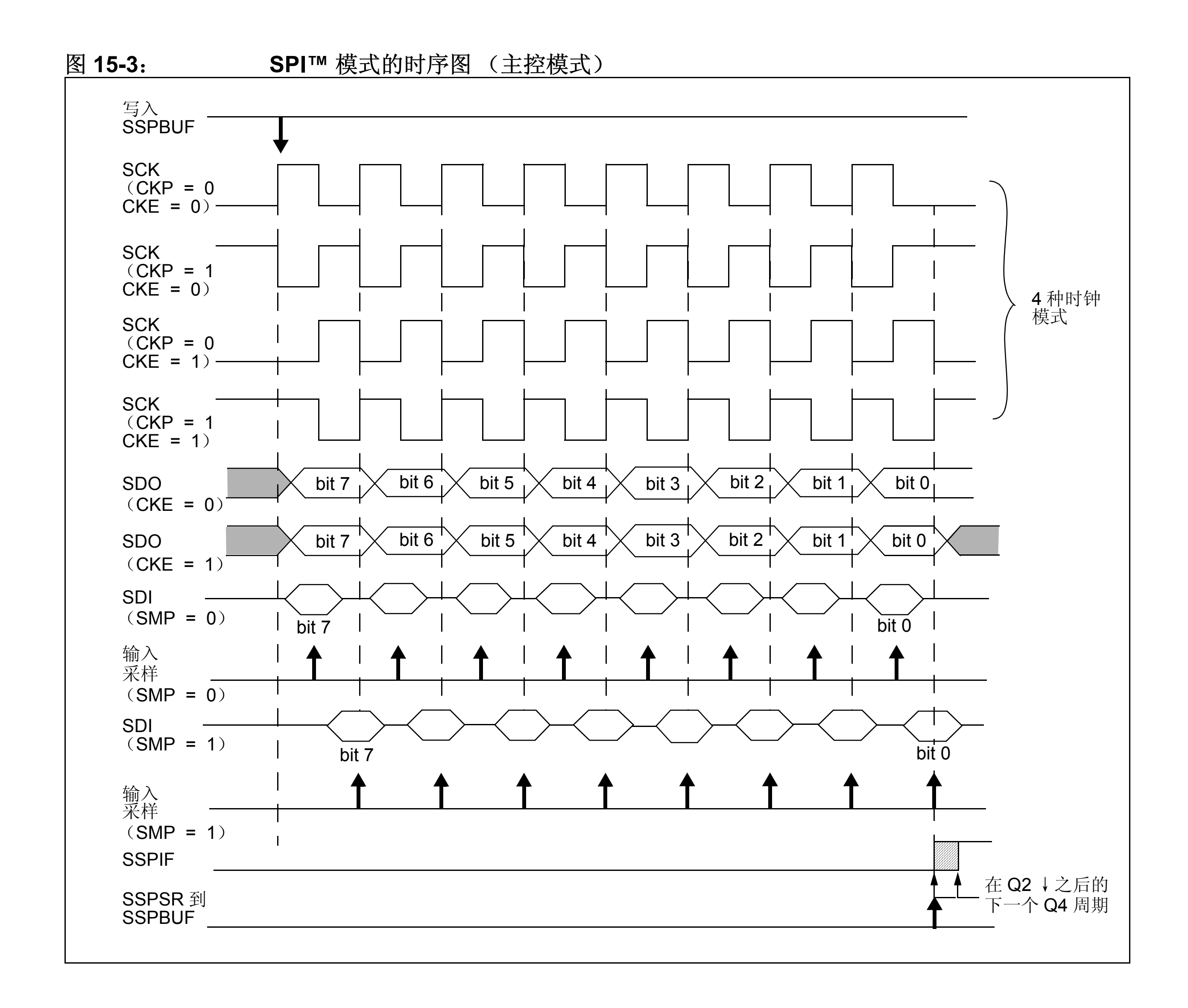
可通过对 CKP 位 （SSPCON<4>）进行适当的编程来选择时钟极性。图 15-3、图 15-5 和图 15-6 将给出 SPI 通信的时序图，其中最先发送的是最高有效位。在主控模式下， SPI 时钟速率（位速率）可由用户编程设定为下面几种之一：

- FOSC/4 （或 TCY）
- FOSC/16 （或 4 TCY）
- FOSC/64 （或 16 TCY）
- Timer2 输出 /2

这样可使数据速率最高达到 10.00 Mbps （时钟频率为 40 MHz）。

图 15-3 给出了主控模式的波形图。当 CKE 位置 1 时，SDO 数据在 SCK 出现时钟边沿前一直有效。图中所示的输入采样的变化由 SMP 状态位反映。图中给出了将接收到的数据装入 SSPBUF 的时间。

图 15-3： SPI™ 模式的时序图 （主控模式）



初稿

15.3.6 从动模式

在从动模式下，当 SCK 引脚上有外部时钟脉冲时启动发送和接收数据。当最后一位数据被锁存后，中断标志位 SSPIF 置 1。

在从动模式下，外部时钟由 SCK 引脚上的外部时钟源提供。外部时钟必须满足电气规范中规定的高电平和低电平的最短时间要求。

在休眠模式下，从器件仍可发送 / 接收数据。当接收到一个字节时，器件从休眠状态中唤醒。

15.3.7 从动选择同步

$\overline{SS}$ 引脚允许器件工作于同步从动模式。SPI 必须处于从动模式，并使能 $\overline{SS}$ 引脚控制（SSPCON1<3:0> = 04h）。要让 $\overline{SS}$ 引脚充当输入端，则不能将其驱动为低电平。数据锁存器必须为高电平。当 $\overline{SS}$ 引脚为低电平时，使能数据的发送和接收，同时驱动 SDO 引脚。当 $\overline{SS}$ 引脚变为高电平时，即使是在字节的发送过程中，也不再驱动 SDO 引脚，而是将其变成高阻悬空状态。根据应用的需要，可在SDO引脚上外接上拉/下拉电阻。

> **注 1：** 当 SPI 处于从动模式，并且 $\overline{SS}$ 引脚控制使能（SSPCON<3:0> = `0100`）时，如果 $\overline{SS}$ 引脚置为 VDD 电平将使 SPI 模块复位。
>
> **2：** 如果 SPI 工作在从动模式下并且 CKE 置 1，则必须使能 $\overline{SS}$ 引脚控制。

当 SPI 模块复位后，位计数器被强制为 `0`。这是通过强制将 $\overline{SS}$ 引脚拉为高电平或将 SSPEN 位清零来实现的。

将 SDO 引脚和 SDI 引脚相连，可以仿真二线制通信。当 SPI 需要作为接收器工作时， SDO 引脚可以被配置为输入端。这样就禁止了从 SDO 发送数据。因为 SDI 不会引起总线冲突，所以可以一直将其保留为输入（SDI 功能）。

图 15-4： 从动同步模式波形

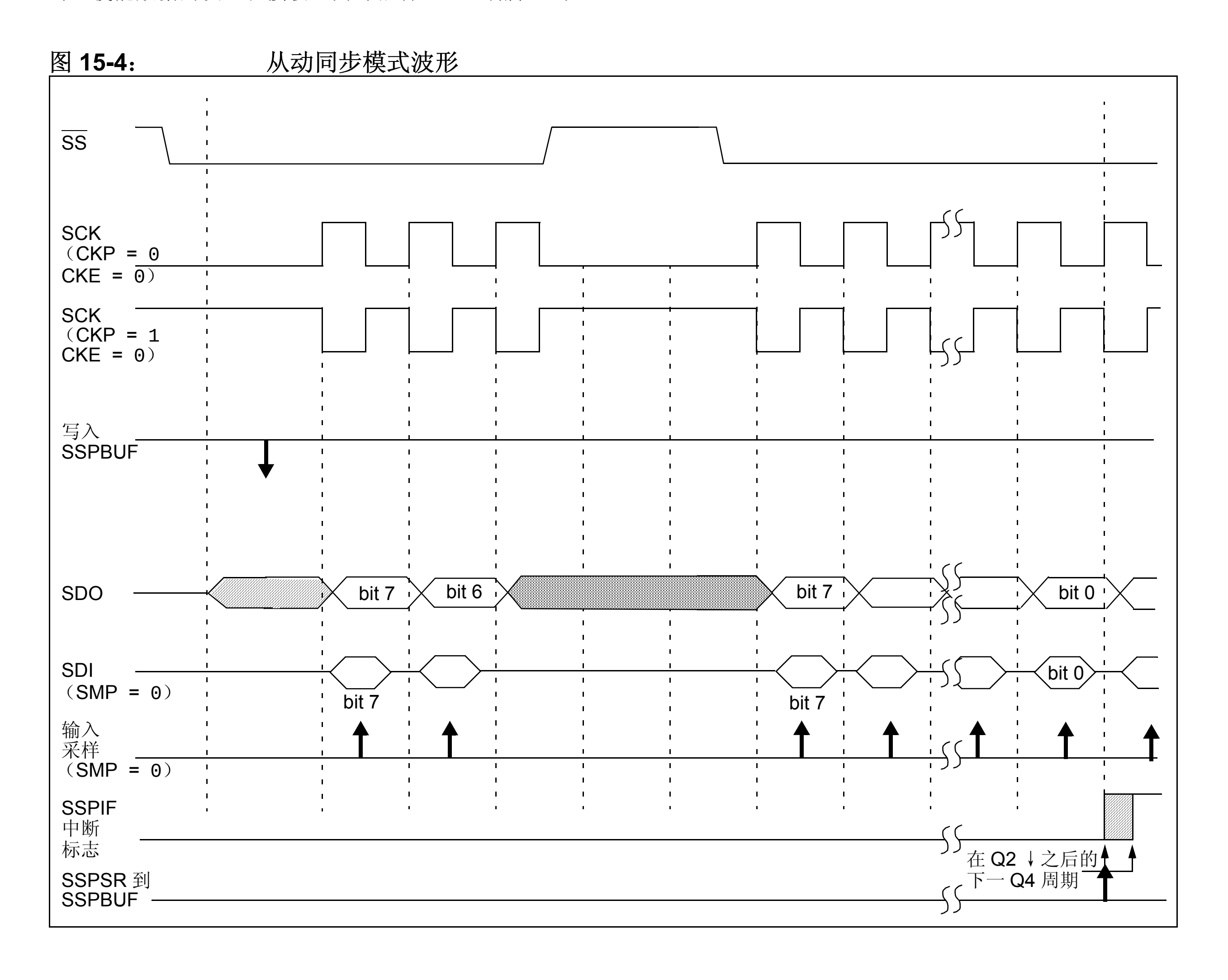


图 15-5： SPI™ 模式的时序图 （从动模式且 CKE = 0）

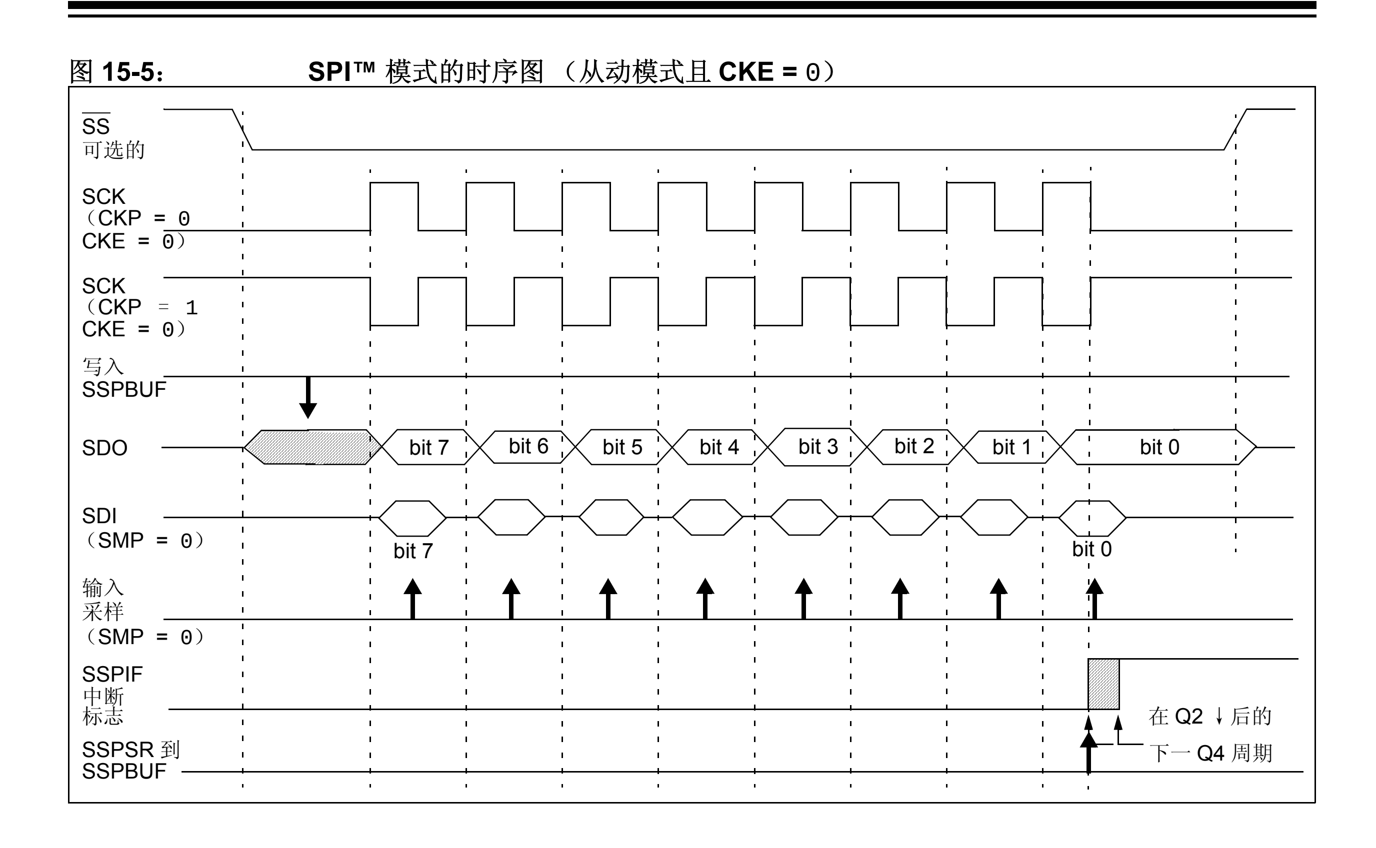
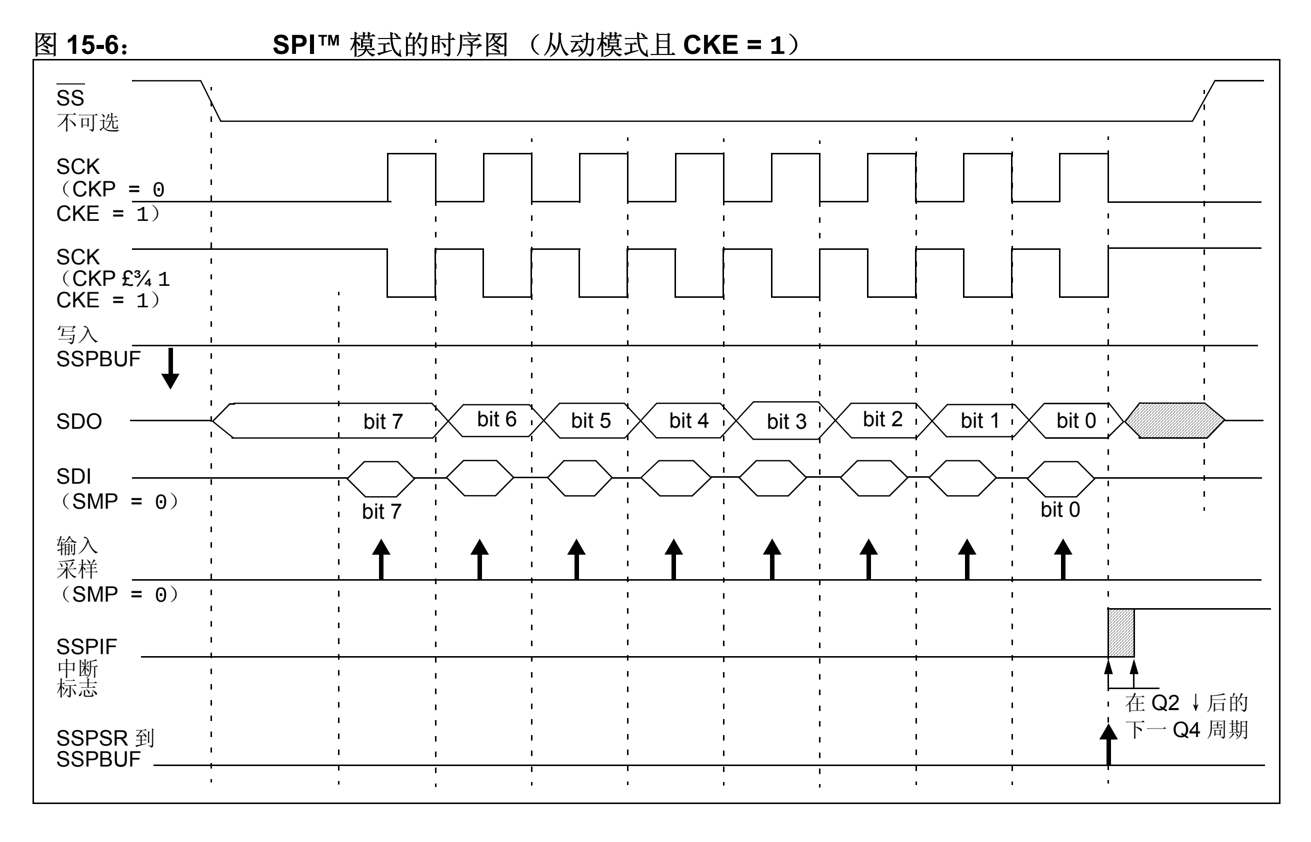


图 15-6： SPI™ 模式的时序图 （从动模式且 CKE = 1）



 初稿

15.3.8 休眠工作方式

在 SPI 主控模式下，模块时钟速度与全功耗模式下的不同；处于休眠模式时，所有时钟都停止。

在大多数功耗管理模式下，为外设提供一个时钟。该时钟应该来自于主时钟源、辅助时钟源 （32.768 kHz 的 Timer1 振荡器）或 INTOSC 时钟源。更多信息请参见**第 2.7 节 "时钟源与振荡器切换"**。

在大多数情况下，主器件为 SPI 数据提供的时钟速度并不重要；但是，每个系统都应该评估此因素。

如果使能了 MSSP 中断，那么当主器件发送完数据时这些中断可以将控制器从休眠模式或某种空闲模式唤醒。如果不想从休眠或空闲模式退出，应该禁止 MSSP 中断。

如果选择了休眠模式，所有模块的时钟都将停止，并且在器件被唤醒前，发送 / 接收将保持此停滞状态。当器件返回到运行模式后，该模块将恢复发送和接收数据。

在 SPI 从动模式下， SPI 发送 / 接收移位寄存器与器件异步工作。这可以使器件处于任何功耗管理模式下，而且数据仍可被移入 SPI 发送 / 接收移位寄存器。当 8 位数据全部接收到后，MSSP 中断标志位将置 1，并且如果中断使能的话，器件被唤醒。

15.3.9 复位的影响

复位操作会禁止 MSSP 模块并终止当前的数据传输。

15.3.10 总线模式兼容性

表 15-1 中所示是标准 SPI 模式与 CKP 和 CKE 控制位状态的对应关系。

表 15-1： SPI 总线模式

标准 SPI™ 模式术语	控制位状态	
	CKP	**CKE**
0, 0	0	1
0, 1	0	0
1, 0	1	1
1, 1	1	0

还有一个 SMP 位用来控制数据何时被采样。

表 15-2： 与 SPI™ 操作相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在的页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
PIR1	—	ADIF	RC1IF	TX1IF	SSPIF	CCP1IF	TMR2IF	TMR1IF	61
PIE1	—	ADIE	RC1IE	TX1IE	SSPIE	CCP1IE	TMR2IE	TMR1IE	61
IPR1	—	ADIP	RC1IP	TX1IP	SSPIP	CCP1IP	TMR2IP	TMR1IP	61
TRISC	PORTC 数据方向寄存器								62
TRISF	PORTF 数据方向寄存器								62
SSPBUF	同步串口接收缓冲器 / 发送寄存器								60
SSPCON1	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	60
SSPSTAT	SMP	CKE	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF	60

图注： — = 未用位，读为 0。 SPI 模式下的 MSSP 不使用阴影单元。

15.4 I²C 模式

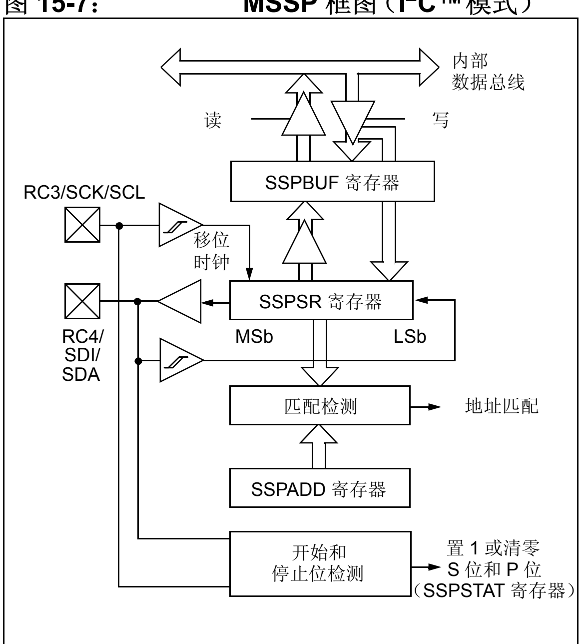
MSSP 模块工作在 I²C 模式时，可以实现所有的主控和从动功能 （包括支持广播呼叫地址），并且硬件上提供启动位和停止位的中断来判断总线何时空闲（多主器件功能）。MSSP 模块实现标准模式规范以及 7 位和 10 位寻址。

有两个引脚用于数据传输：

- 串行时钟 （SCL）－ RC3/SCK/SCL
- 串行数据 （SDA）－ RC4/SDI/SDA

用户必须通过将 TRISC<4:3> 位置 1 将上述引脚配置为输入引脚。

图 15-7： MSSP 框图（I²C™ 模式）



15.4.1 寄存器

MSSP 模块有 6 个寄存器用于 I²C 操作。这些寄存器包括：

- MSSP 控制寄存器 1 （SSPCON1）
- MSSP 控制寄存器 2 （SSPCON2）
- MSSP 状态寄存器 （SSPSTAT）
- 串行接收 / 发送缓冲寄存器 （SSPBUF）
- MSSP 移位寄存器 （SSPSR）－不能直接访问
- MSSP 地址寄存器 （SSPADD）

SSPCON1、SSPCON2 和 SSPSTAT 是在 I²C 模式下的控制寄存器和状态寄存器。 SSPCON1 和 SSPCON2 寄存器是可读写的。SSPSTAT 的低六位是只读的，而高两位是可读写的。

SSPSR 是用来将数据移入或移出的移位寄存器。SSPBUF 是缓冲寄存器，用于数据字节的写入或读出。

在 I²C 从动模式下配置 SSP 时， SSPADD 寄存器将保存从器件的地址。在主控模式下配置 SSP 时，SSPADD 的低 7 位用作波特率发生器的重载值。

接收数据时，SSPSR 和 SSPBUF 共同构成一个双重缓冲接收器。当 SSPSR 接收到一个完整的字节后，该字节被送入 SSPBUF 寄存器，同时将中断标志位 SSPIF 置 1。

在发送过程中， SSPBUF 并不是双重缓冲的。对 SSPBUF 的写操作将同时写入 SSPBUF 和 SSPSR。

 初稿

寄存器 15-3： SSPSTAT：MSSP 状态寄存器 （I²C™ 模式）

R/W-0	R/W-0	R-0	R-0	R-0	R-0	R-0	R-0
SMP	CKE	D/$\overline{A}$	P[(1)]	S[(1)]	R/$\overline{W}$[(2,3)]	UA	BF
bit 7							bit 0

bit 7 **SMP**：变化率控制位
在主控或从动模式下：
1 = 标准速度模式下禁止变化率控制 （100 kHz 和 1 MHz）
0 = 高速模式下使能变化率控制 （400KHz）

bit 6 **CKE**：SMBus 选择位
在主控或从动模式下：
1 = 使能 SMBus 特定输入
0 = 禁止 SMBus 特定输入

bit 5 **D/$\overline{A}$**：数据 / $\overline{地址}$标志位
在主控模式下：
保留。
在从动模式下：
1 = 表示上一个接收或发送的字节是数据
0 = 表示上一个接收或发送的字节是地址

bit 4 **P**：停止位 [(1)]
1 = 表示最近检测到停止位
0 = 表示最近未检测到停止位

bit 3 **S**：启动位 [(1)]
1 = 表示最近检测到起始位
0 = 表示最近未检测到起始位

bit 2 **R/$\overline{W}$**：读 / $\overline{写}$位信息 （仅用于 I²C 模式）
在从动模式下：[(2)]
1 = 读
0 = 写
在主控模式下：[(3)]
1 = 正在进行发送
0 = 未进行发送

bit 1 **UA**：更新地址位 （仅用于 10 位从动模式）
1 = 表示用户需要更新 SSPADD 寄存器中的地址
0 = 不需要更新地址

bit 0 **BF**：缓冲器满状态位
在发送模式下：
1 = 接收完成， SSPBUF 已满
0 = 接收未完成， SSPBUF 为空
在接收模式下：
1 = 数据正在发送 （不包括 $\overline{ACK}$ 和停止位）， SSPBUF 已满
0 = 数据发送已完成 （不包括 $\overline{ACK}$ 和停止位）， SSPBUF 已空

注 1：当 SSPEN 被清零或复位时该位被清零。
2：该位用来保存在最近一次地址匹配后的 R/$\overline{W}$ 位信息。该位仅在从地址匹配开始到下一个起始位、停止位或非 $\overline{ACK}$ 位期间有效。
3：该位与 SEN、 RSEN、 PEN、 RCEN 或 ACKEN 进行 “或”运算的结果表示 MSSP 是否处于空闲模式。

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

寄存器 15-4： SSPCON1：MSSP 控制寄存器 1 （I²C 模式）

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
WCOL	SSPOV	SSPEN(1)	CKP	SSPM3(2)	SSPM2(2)	SSPM1(2)	SSPM0(2)
bit 7							bit 0

bit 7 **WCOL**：写冲突检测位
在主控发送模式下：
1 = 当 I²C 不满足启动发送数据的条件时，
试图向 SSPBUF 寄存器写入数据 （必须用软件清零）
0 = 未发生冲突
在从动发送模式下：
1 = 正在发送前一个字时，又有数据写入 SSPBUF 寄存器 （必须
用软件清零）
0 = 未发生冲突
在接收模式 （主控或从动模式）下：
在此忽略。

bit 6 **SSPOV**：接收溢出指示位
在接收模式下：
1 = SSPBUF 寄存器仍在保存前一字节时，接收到一个新的字节 （必须用软件清零）
0 = 无溢出
在发送模式下：
在发送模式下，此位忽略。

bit 5 **SSPEN**：同步串口使能位 (1)
1 = 使能串口并将 SDA 和 SCL 引脚配置为串口引脚
0 = 禁止串口并将上述引脚配置为 I/O 端口引脚

bit 4 **CKP**：SCK 释放控制位
在从动模式下：
1 = 释放时钟
0 = 保持时钟低电平 （延长低电平时间），用来确保数据建立
在主控模式下：
在此模式下未使用。

bit 3-0 **SSPM3:SSPM0**：同步串口模式选择位 (2)
1111 = I²C 从动模式， 10 位地址，并使能启动位和停止位中断
1110 = I²C 从动模式， 7 位地址，并使能启动位和停止位中断
1011 = I²C 由固件控制的主控模式 （从器件空闲）
1000 = I²C 主控模式，时钟 = FOSC/(4 * (SSPADD + 1))
0111 = I²C 从动模式， 10 位地址
0110 = I²C 从动模式， 7 位地址

注 **1**：当该位被使能时，必须将 SDA 和 SCL 引脚配置为输入或输出引脚。
2：此处未列出的位组合为保留或只用于 SPI 模式。

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

寄存器 15-5： SSPCON2：MSSP 控制寄存器 2 （I²C™ 模式）

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
GCEN	ACKSTAT	ACKDT(1)	ACKEN(2)	RCEN(2)	PEN(2)	RSEN(2)	SEN(2)
bit 7							bit 0

bit 7 **GCEN**：广播呼叫使能位 （仅用于从动模式）
1 = 当 SSPSR 接收到广播呼叫地址 （0000h）时使能中断
0 = 禁止广播呼叫地址

bit 6 **ACKSTAT**：应答状态位 （仅用于主控发送模式）
1 = 未收到来自从器件的应答
0 = 收到来自从器件的应答

bit 5 **ACKDT**：应答数据位 （仅用于主控接收模式）(1)
1 = 无应答
0 = 应答

bit 4 **ACKEN**：应答序列使能位 （仅用于主控接收模式）(2)
1 = 在 SDA 和 SCL 引脚上发起应答序列，并发送 ACKDT 数据位。
由硬件自动清零。
0 = 应答序列空闲

bit 3 **RCEN**：接收使能位 （仅用于主控模式）(2)
1 = 使能 I²C 接收模式
0 = 接收空闲

bit 2 **PEN**：停止条件使能位 （仅用于主控模式）(2)
1 = 在 SDA 和 SCL 引脚上发起停止条件。由硬件自动清零。
0 = 停止条件空闲

bit 1 **RSEN**：重复启动条件使能位 （仅用于主控模式）(2)
1 = 在 SDA 和 SCL 引脚上发起重复启动条件。由硬件自动清零
0 = 重复启动条件空闲

bit 0 **SEN**：启动条件使能 / 延长使能位 (2)
在主控模式下：
1 = 在 SDA 和 SCL 引脚上发起启动条件。由硬件自动清零。
0 = 启动条件空闲
在从动模式下：
1 = 为从动发送和从动接收 （已使能延长）使能时钟低电平时间延长
0 = 时钟低电平时间延长被禁止。

注 **1**：用户在接收结束时发起一个应答时序，同时发送该值。
2：如果 I²C 模块不处于空闲模式，该位不能被置 1 （不支持并行操作），而且不能对 SSPBUF 进行写操作 （或者禁止写 SSPBUF）。

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

15.4.2 工作原理

通过将 MSSP 使能位 SSPEN（SSPCON1<5>）置 1，可使能 MSSP 模块。

SSPCON1 寄存器用于控制 I^{2}C 工作模式。可通过设置模式选择位（SSPCON1<3:0>）选择以下 I^{2}C 模式之一：

- I^{2}C 主控模式，时钟 = （Fosc/4）×（SSPADD + 1）
- I^{2}C 从动模式 （7 位地址）
- I^{2}C 从动模式 （10 位地址）
- I^{2}C 从动模式 （7 位地址），使能启动位和停止位中断
- I^{2}C 从动模式 （10 位地址），使能启动位和停止位中断
- I^{2}C 固件控制的主控模式，从器件空闲

通过将相应的 TRISC 位置 1，将 SCL 和 SDA 引脚编程为输入引脚；在 SSPEN 位置 1 时选择任何 I^{2}C 模式，将强制上述引脚漏极开路。要确保此模块的正常工作，必须为 SCL 和 SDA 引脚提供外接上拉电阻。

15.4.3 从动模式

在从动模式下， SCL 引脚和 SDA 引脚必须被配置为输入 （TRISC<4:3> 置 1）。必要时 MSSP 模块将使用输出数据改写输入状态 （从发送器）。

I^{2}C 从动模式硬件总是在地址匹配时产生中断。用户也可以通过模式选择位，选择启动位或停止位中断。

当地址匹配或在地址匹配后发送的数据被接收时，硬件会自动产生一个应答 （$\overline{ACK}$）脉冲，并把当时 SSPSR 寄存器中接收到的值装入 SSPBUF 寄存器。

只要满足下列条件之一，MSSP 模块就不会产生此 $\overline{ACK}$ 脉冲：

- 在接收到数据前，缓冲器满位 BF（SSPSTAT<0>）置 1。
- 在接收到数据之前，溢出位 SSPOV（SSPCON1<6>）被置 1。

在这种情况下，SSPSR寄存器的值不会载入SSPBUF，但是 SSPIF （PIR1<3>）会置 1。 BF 位是通过读取 SSPBUF 寄存器清零的，而 SSPOV 位是通过软件清零的。

为确保正常工作，SCL 时钟输入必须满足最小高电平和最小低电平时间要求。在时序参数 #100 和参数 #101 中显示了 I^{2}C 规范的高低电平时间和对 MSSP 模块的具体要求。

15.4.3.1 寻址

一旦 MSSP 模块被使能，它就会等待启动条件出现。启动条件出现后， 8 位数据被移入 SSPSR 寄存器。在时钟信号（SCL）的上升沿采样所有的输入位。在第 8 个时钟 （SCL）脉冲的下降沿寄存器 SSPSR<7:1> 的值会和 SSPADD 地址寄存器的值比较。如果地址匹配，并且 BF 位和 SSPOV 位都被清零，会发生下列事件：

1. SSPSR 寄存器值被装入 SSPBUF 寄存器。
2. 将缓冲器满标志位 BF 置 1。
3. 产生 $\overline{ACK}$ 脉冲。
4. 在第 9 个 SCL 脉冲的下降沿， MSSP 中断标志位 SSPIF （PIR1<3>）置 1（如果使能中断，则产生中断）。

在 10 位地址模式下，从器件需要接收两个地址字节。第一个地址字节的高 5 位将指定这是否是一个 10 位地址。R/$\overline{W}$ 位（SSPSTAT<2>）必须指定写操作，这样从器件才能接收到第二个地址字节。对于 10 位地址，第一个字节应该是 “`11110 A9 A8 0`”，其中 “`A9`” 和 “`A8`” 是该地址的两个最高有效位。 10 位地址的工作步骤如下，其中 7-9 步是针对从动发送器而言的。

1. 接收地址的第一个 （高）字节 （SSPIF 位、BF 位和 UA 位 （SSPSTAT<1>）置 1）。
2. 用地址的第二个 （低）字节更新 SSPADD 寄存器 （UA 位清零并释放 SCL 时钟线）。
3. 读 SSPBUF 寄存器 （BF 位清零）并将标志位 SSPIF 清零。
4. 接收地址的第二个 （低）字节 （SSPIF 位、BF 位和 UA 位置 1）。
5. 使用地址的第一个 （高）字节更新 SSPADD 寄存器。如果匹配的话就释放 SCL 时钟线，这将清零 UA 位。
6. 读 SSPBUF 寄存器 （BF 位清零）并将标志位 SSPIF 清零。
7. 接收重复启动条件。
8. 接收地址的第一个 （高）字节 （SSPIF 位和 BF 位置 1）。
9. 读 SSPBUF 寄存器 （BF 位清零）并将标志位 SSPIF 清零。

15.4.3.2 接收

当地址字节的 R/W 位清零并发生地址匹配时，SSPSTAT 寄存器的 R/W 位清零。接收的地址被装入 SSPBUF 寄存器，且 SDA 信号保持低电平 （ACK）。

当发生地址字节溢出时，则不会产生应答脉冲（ACK）。溢出条件是指 BF 位（SSPSTAT<0>）置 1，或者 SSPOV 位 （SSPCON1<6>）置 1。

每个数据传输字节都会产生一个 MSSP 中断。标志位 SSPIF （PIR1<3>）必须用软件清零。通过 SSPSTAT 寄存器可以确定该字节的状态。

如果 SEN 被使能（SSPCON2<0> = 1）， RC3/SCK/SCL 将在每个数据传输之后保持为低电平（低电平时间延长）。必须通过将 CKP 位（SSPCON1<4>）置 1 才能释放时钟。更多详情请参见**第 15.4.4 节“时钟延长”**。

15.4.3.3 发送

当输入的地址字节的 R/W 位置 1 并发生地址匹配时， SSPSTAT 寄存器的 R/W 位置 1。接收到的地址被装入 SSPBUF 寄存器。 ACK 脉冲在第 9 位上发送，同时不管 SEN 的值如何，RC3/SCK/SCL 引脚保持低电平（如需了解更多细节，请参见**第 15.4.4 节 “时钟延长”**）。通过延长时钟低电平时间，主器件只有在从器件准备好发送数据时，才发出另一个时钟脉冲。发送的数据必须被装入 SSPBUF 寄存器，同时也被装入 SSPSR 寄存器。然后，应该通过将 CKP （SSPCON1<4>）置 1 来使能 RC3/SCK/SCL 引脚。8 个数据位在 SCL 时钟输入的下降沿被移出。这可确保在 SCL 为高电平期间 SDA 信号是有效的 （图 15-9）。

来自主接收器的 ACK 脉冲将在第 9 个 SCL 输入脉冲的上升沿锁存。若 SDA 信号为高电平 （无 ACK 应答信号），那么表示数据传输已完成。在这种情况下，如果从器件锁存了 ACK，将复位从动逻辑（复位 SSPSTAT 寄存器），同时从器件监视下一个起始位的出现。如果 SDA 线为低电平（ACK），则必须将下一个要发送的数据装入 SSPBUF 寄存器。同样，必须通过将 CKP 位置 1 来使能 RC3/SCK/SCL 引脚。

每个数据传输字节都会产生一个 MSSP 中断。SSPIF 位必须用软件清零， SSPSTAT 寄存器用于确定字节的状态。 SSPIF 位在第 9 个时钟脉冲的下降沿被置 1。

图 15-8： **I²C™ 从动模式的接收时序 （SEN = 0， 7 位地址）**

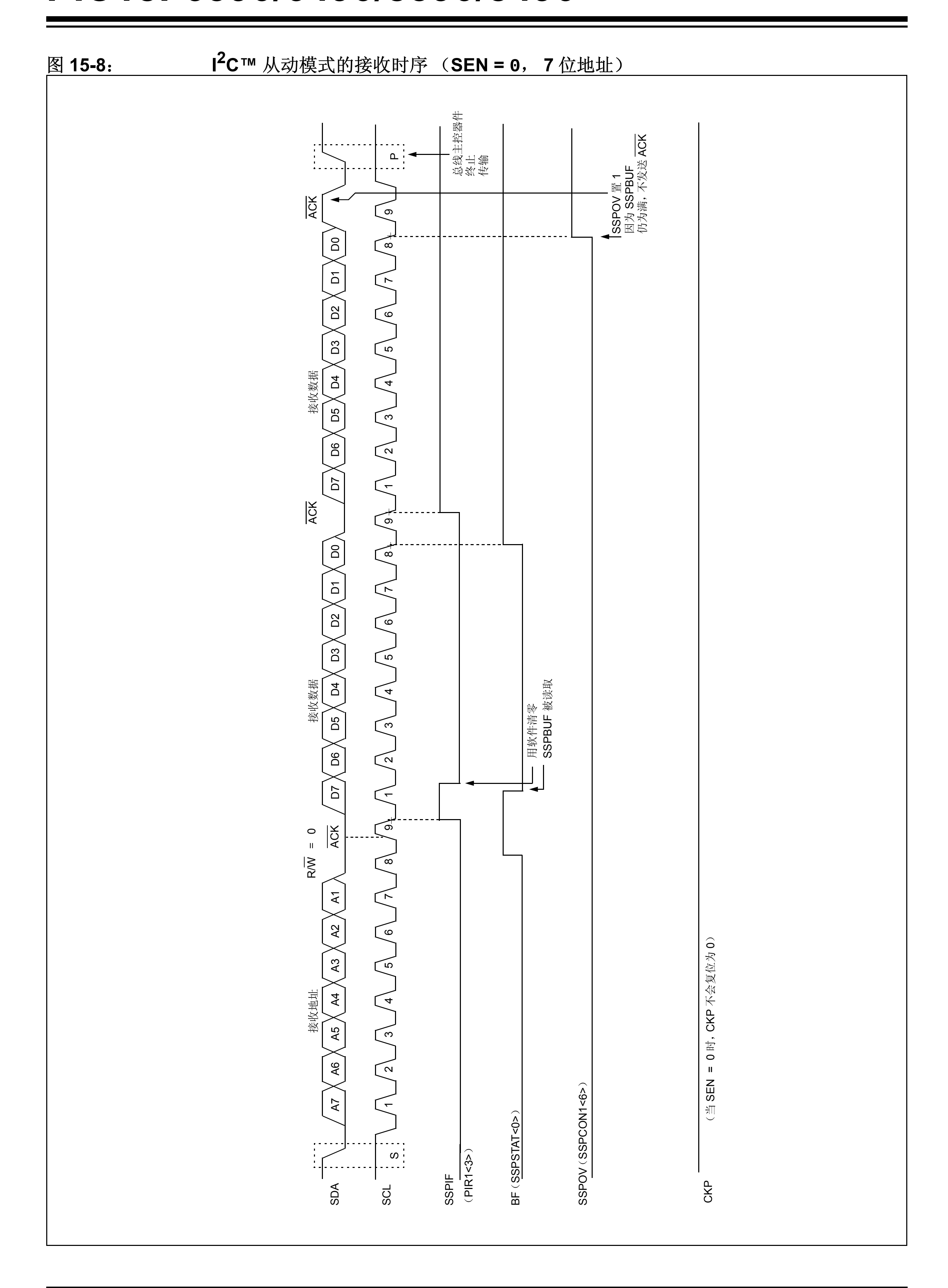


图 15-9： I²C™ 从动模式发送时序 （7 位地址）

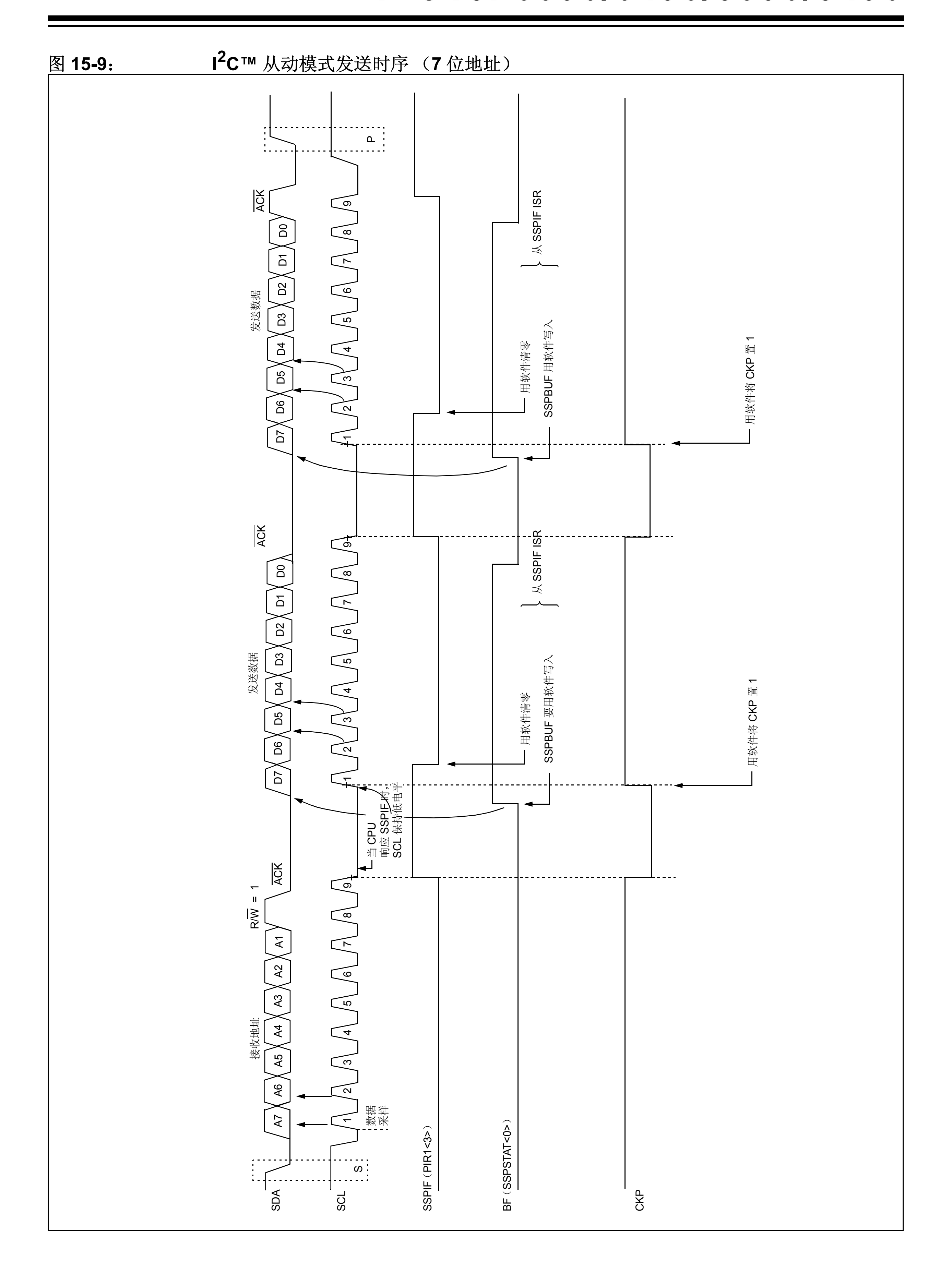
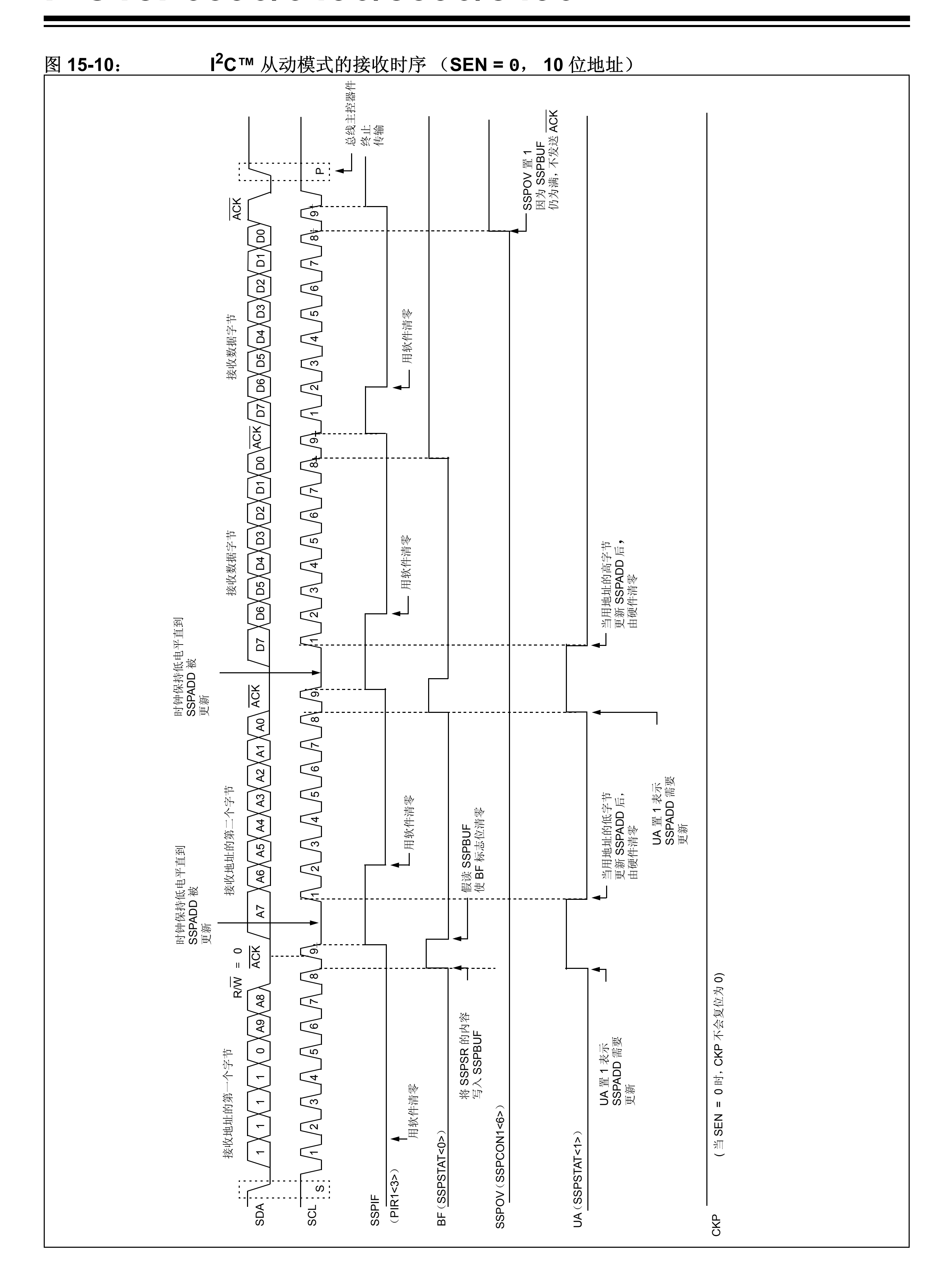
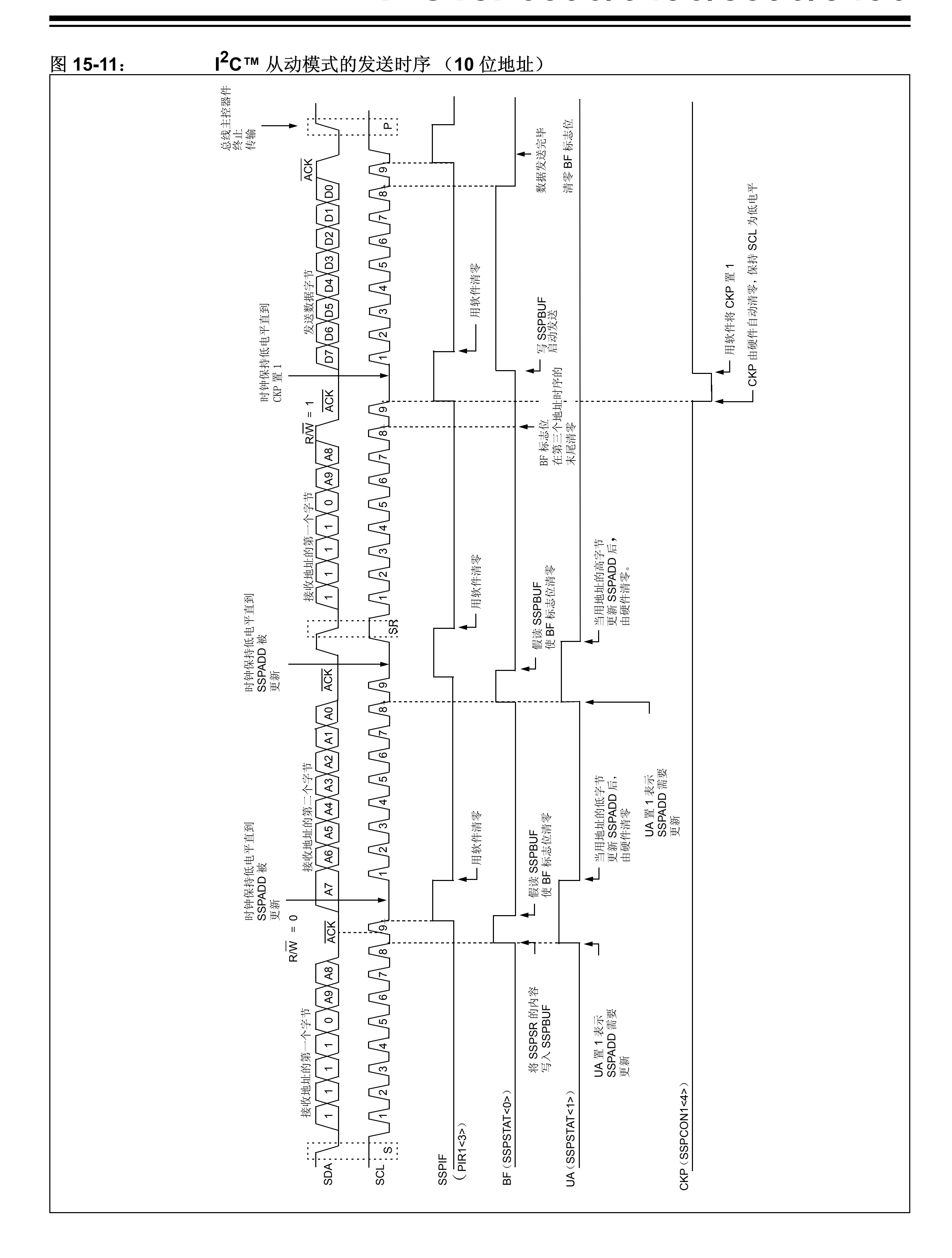


图 15-10： I²C™ 从动模式的接收时序 （SEN = 0， 10 位地址）



 初稿

图 15-11： I²C™ 从动模式的发送时序 （10 位地址）



15.4.4 时钟延长

7位和10位从动模式都在发送序列中实现了自动时钟延长。

SEN 位 （SSPCON2<0>）允许在接收过程中使能时钟延长。将 SEN 置 1 会导致在每个数据接收序列的末尾将 SCL 引脚保持在低电平。

15.4.4.1 7 位从动接收模式的时钟延长（SEN = 1）

在 7 位从动接收模式下，如果在 $\overline{ACK}$ 序列末的第 9 个时钟的下降沿将 BF 位置 1，则 SSPCON1 寄存器中的 CKP 位就会自动清零，强制 SCL 输出保持在低电平。CKP 被清零会将 SCL 线拉为低电平。在允许继续接收之前，必须在用户的中断服务程序中将 CKP 位置 1。保持SCL信号为低电平器件让用户在主器件发起另一个接收序列之前，有时间处理中断服务程序并读取 SSPBUF 的内容。这将防止发生缓冲器溢出 （见图 15-13）。

注	
1：	如果用户在第 9 个时钟的下降沿到来之前读取了 SSPBUF 的内容，使得 BF 位被清零，那么 CKP 位就不会被清零，也不会发生时钟延长。
2：	不管 BF 位的状态如何， CKP 位都可以用软件置 1。在下一个接收序列开始之前，用户要注意在中断服务程序中清零BF位时，以避免溢出。

15.4.4.2 10 位从动接收模式的时钟延长（SEN = 1）

在 10 位从动接收模式下，在地址序列中会自动发生时钟延长，但是 CKP 位不会被清零。在这期间，如果 UA 位在第 9 个时钟之后置 1，将启动时钟延长。UA 位在接收到 10 位地址的高字节后被置 1，然后接收 10 位地址的第二个字节并清零 R/$\overline{W}$ 位。在更新 SSPADD 的时候释放时钟线。如同 7 位模式一样，在每个数据接收序列中会发生时钟延长。

注：	如果用户在第9个时钟的下降沿出现之前查询 UA 位，并通过更新 SSPADD 寄存器清零 UA 位，而且在此之前用户没有读取 SSPBUF 寄存器使 BF 位清零，则 CKP 位的电平仍然不会被拉低。基于 BF 位状态的时钟延长仅在数据序列中出现，不会出现在地址序列中。

15.4.4.3 7 位从动发送模式的时钟延长

如果 BF 位被请零，7 位从动发送模式将通过在第 9 个时钟的下降沿出现后清零 CKP 位，以实现时钟延长。上述情形与 SEN 位的状态无关。

用户的中断服务程序必须先将 CKP 位置 1 才可以继续发送。在保持 SCL 信号为低电平期间，用户在主器件发起另一个发送序列之前，将有时间处理中断服务程序并装入 SSPBUF 的内容 （见图 15-9）。

注	
1：	如果用户在第 9 个时钟的下降沿之前就装入 SSPBUF 的内容， CKP 位就不会被清零，也不会发生时钟延长。
2：	不管 BF 位的状态如何， CKP 位都可以用软件置 1。

15.4.4.4 10 位从动发送模式的时钟延长

在 10 位从动发送模式下，在前两个地址序列中由 UA 位的状态来控制时钟延长，正如同在 10 位从动接收模式一样。头两个地址后跟着第三个地址序列，该地址序列包含 10 位地址的高位和被置为 1 的 R/$\overline{W}$ 位。在执行完第三个地址序列后，UA 位不置 1，此时模块配置为发送模式， BF 标志位控制时钟延长，正如 7 位从动发送模式一样 （见图 15-11）。

15.4.4.5 时钟同步和 CKP 位

当 CKP 位被清零时， SCL 输出被强制为 0。然而，将 CKP 位置 1 不会将 SCL 输出拉为低电平，除非已经采样到 SCL 输出为低电平。因此，CKP 位不会将 SCL 信号拉为低电平，除非外部 I²C 主器件将 SCL 线拉低。SCL 输出将保持低电平，直到 CKP 位置 1 且 I²C 总线上的其他器件将SCL电平拉高为止。这可以确保对CKP 位的写操作不会违反 SCL 的最小高电平时间要求（见图 15-12）。

图 15-12： 时钟同步时序

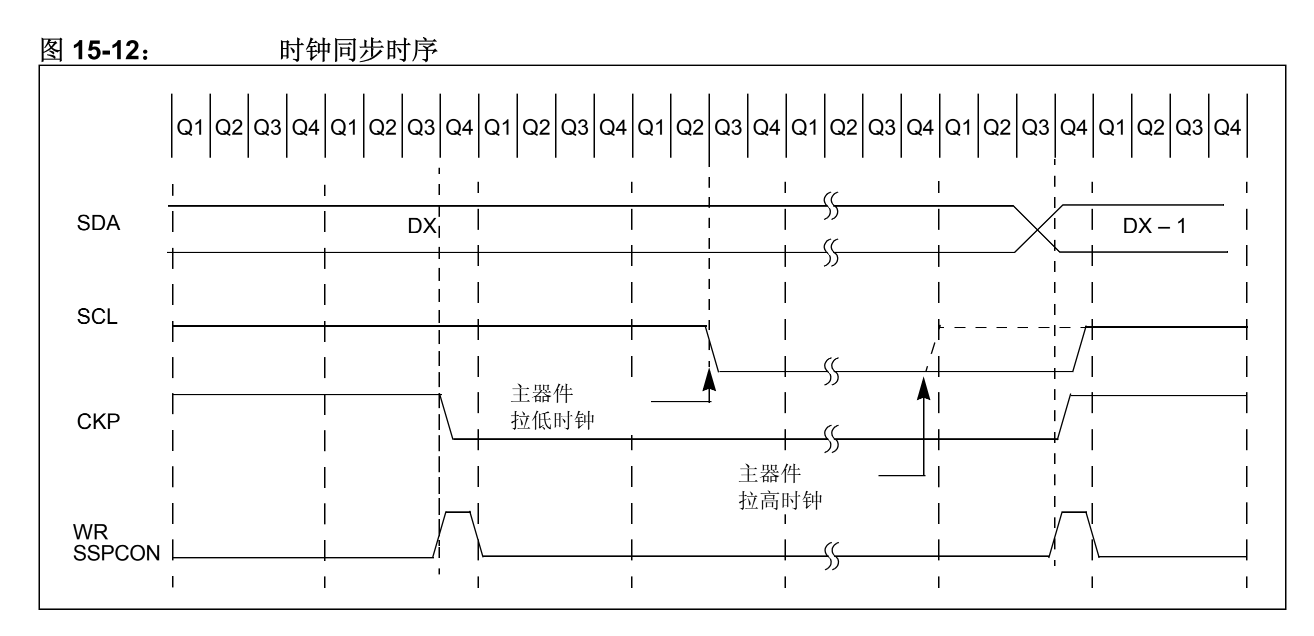
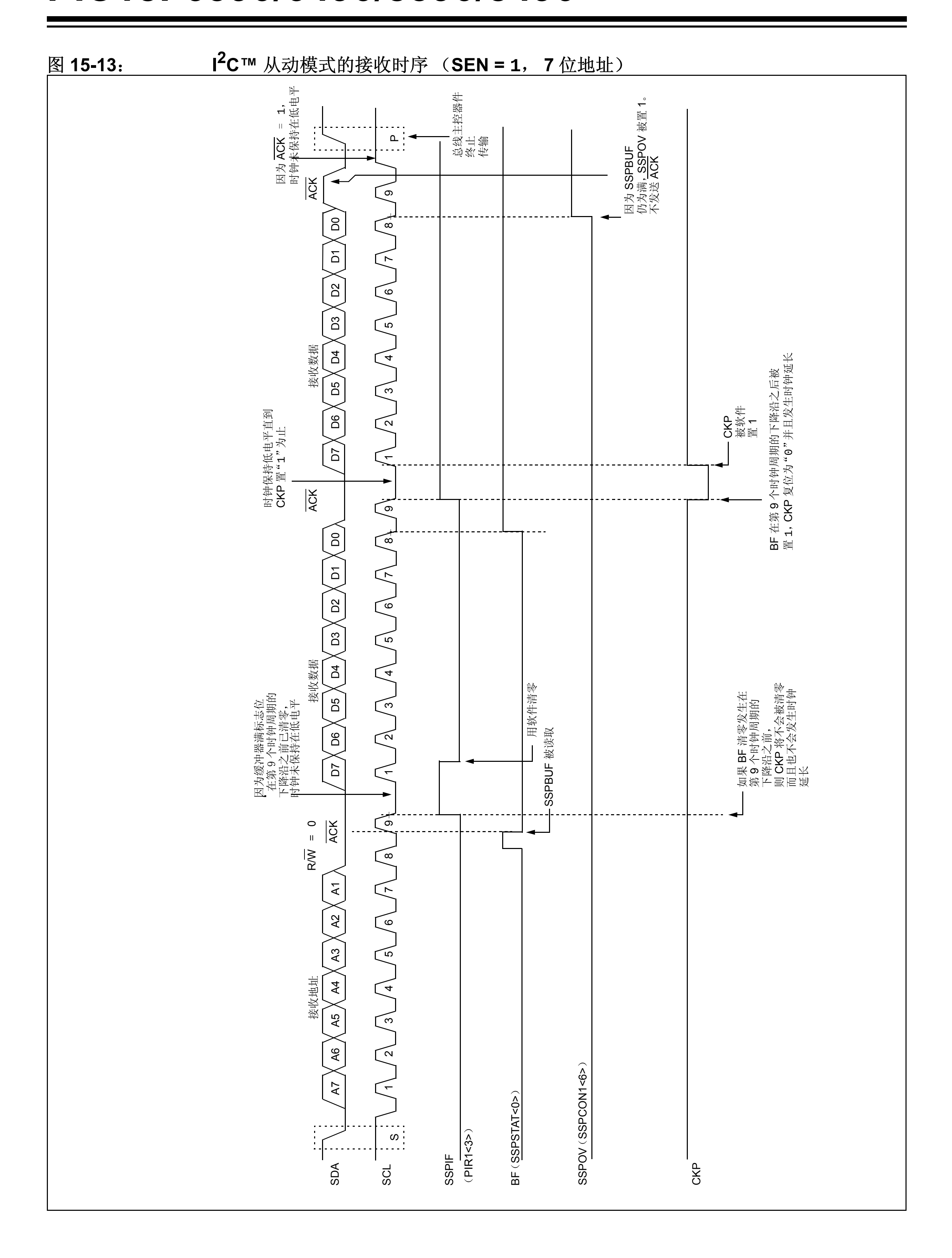
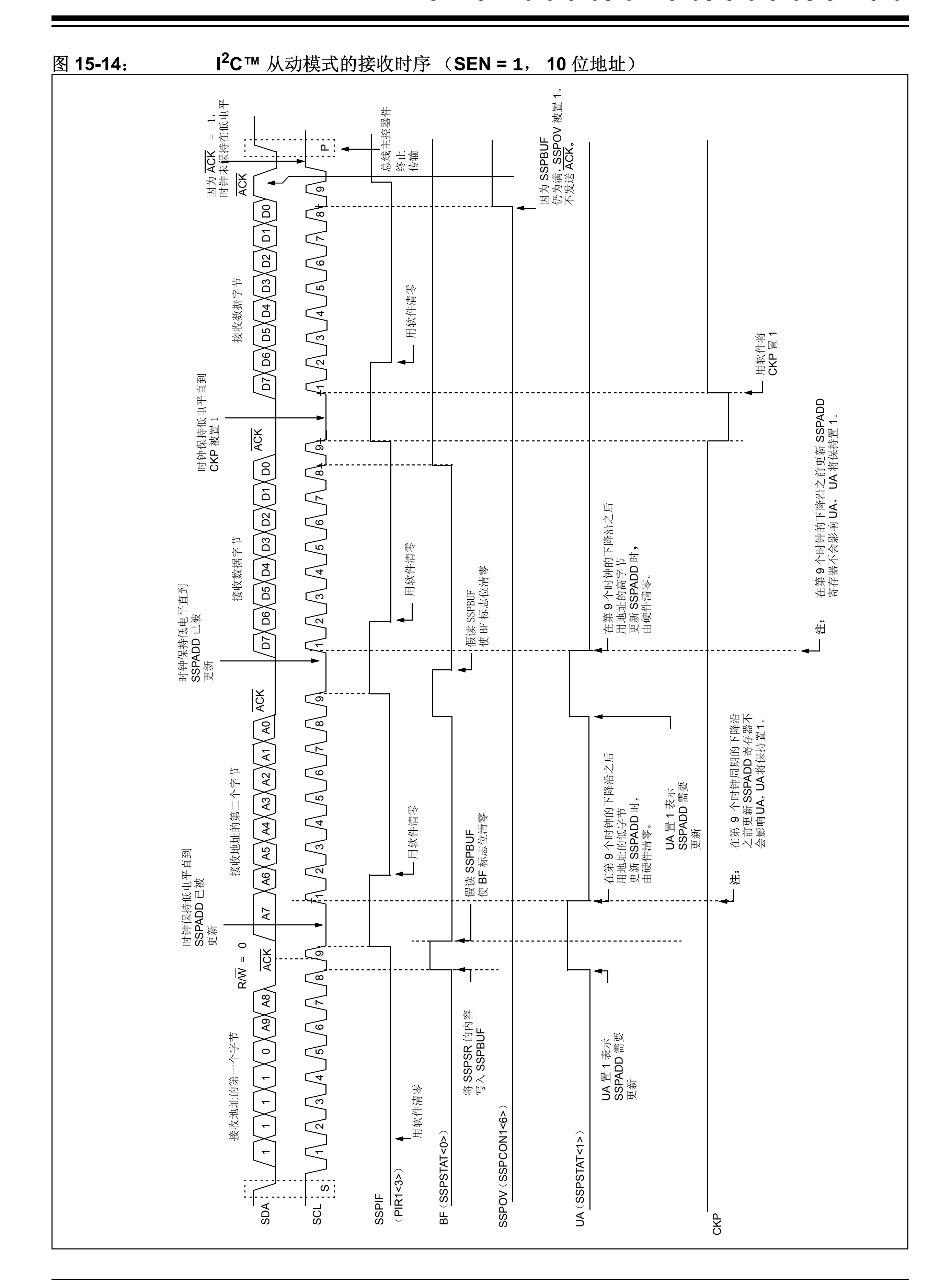


图 15-13： I²C™ 从动模式的接收时序 （SEN = 1， 7 位地址）



 初稿

图 15-14： I^2C™ 从动模式的接收时序（SEN = 1， 10 位地址）



15.4.5 支持广播呼叫地址

在 I²C 总线的寻址过程中，通常由启动条件后的第一个字节决定主器件将寻址哪个从器件。但广播呼叫地址例外，它能寻址所有器件。当使用这个地址时，理论上所有的器件都应该发送一个应答信号来响应。

广播呼叫地址是由 I²C 协议为特定目的保留的 8 个地址之一。它由全 “0” 组成，且 $R/\overline{W}$ = 0。

当使能广播呼叫使能位 （GCEN）（SSPCON2<7> 置 1）时，即可识别广播呼叫地址。检测到启动位后，8 位数据会被移入 SSPSR，同时将该地址与 SSPADD 进行比较。它还会与广播呼叫地址进行比较并用硬件设定。

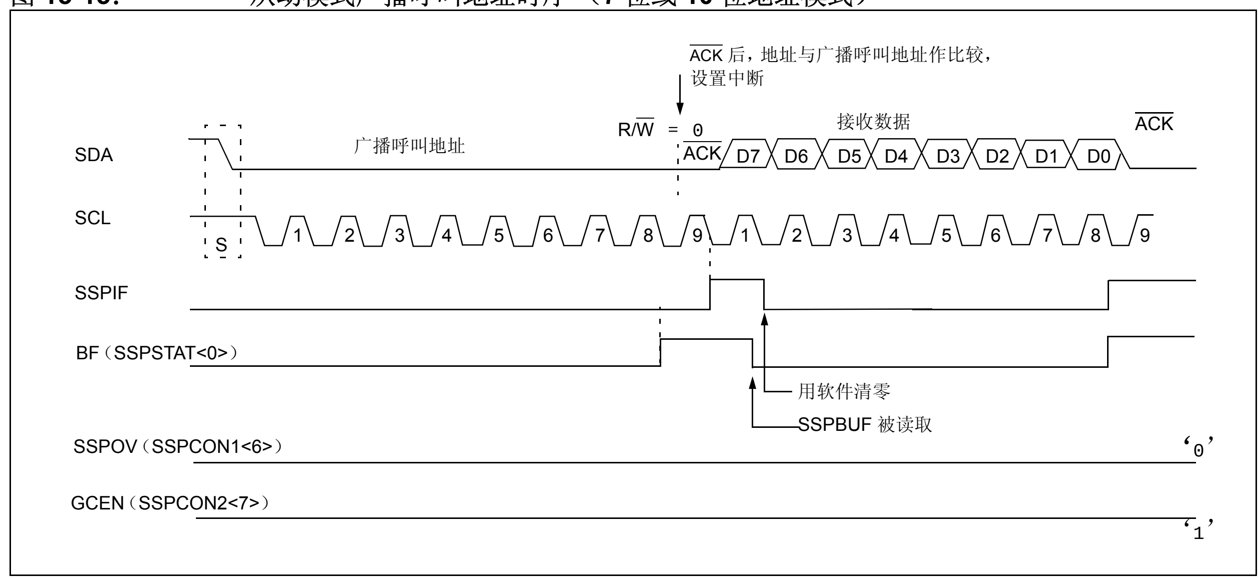
如果与广播呼叫地址匹配， SSPSR 的值将被传输到 SSPBUF，BF 标志位（第 8 位）置 1，并且 SSPIF 中断标志位在第 9 位 （$\overline{ACK}$ 位）的下降沿置 1。

当中断得到响应时，可以通过读取 SSPBUF 的内容来检查中断源。该值可以用于判断是特定器件的地址还是一个广播呼叫地址。

在 10 位模式下，需要更新 SSPADD 用来匹配地址的后半部分，同时 UA 位置 1（SSPSTAT<1>）。如果 GCEN 位置 1 时采样到广播呼叫地址，同时从器件被配置为 10 位地址模式，则不再需要地址的后半部分，也不会将 UA 位置 1，从器件将在应答后开始接收数据（图 15-15）。

图 15-15： 从动模式广播呼叫地址时序 （7 位或 10 位地址模式）





15.4.6 主控模式

通过将 SSPCON1 中的相应 SSPM 位置 1 和清零，同时将 SSPEN 位置 1，可以使能主控模式。在主控模式下， SCL 和 SDA 信号线由 MSSP 硬件控制。

主控模式通过在检测到启动和停止条件时产生中断来工作。停止（P）位和启动（S）位在复位时或禁止 MSSP 模块时清零。当 P 位置 1 时，可以获得 I^{2}C 总线的控制权；否则， P 位和 S 位都清零，总线处于空闲状态。

在固件控制的主控模式下，用户代码根据起始和停止位信号执行所有的 I^{2}C 总线操作。

一旦使能主控模式，用户即可选择以下 6 项操作：

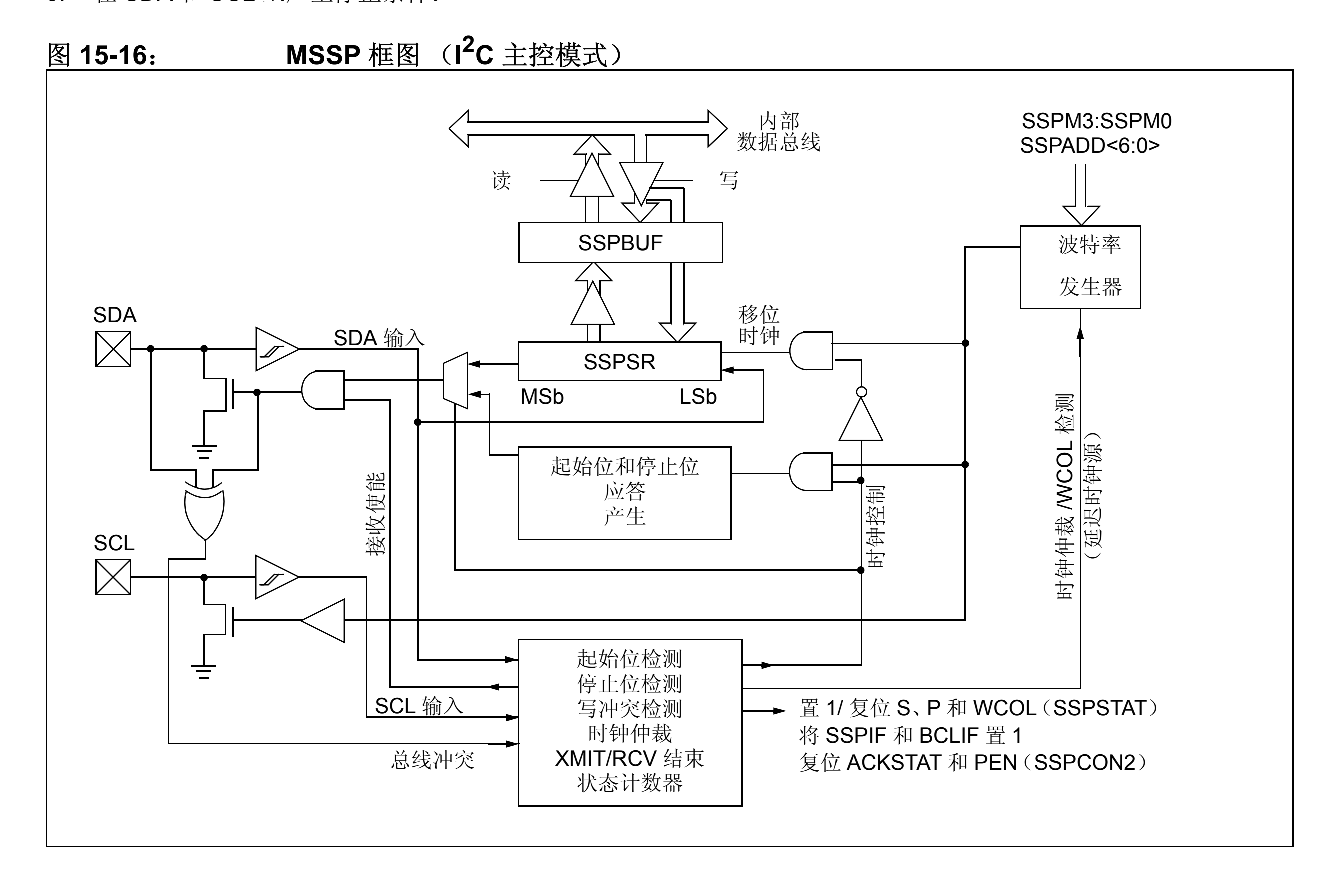
1. 在 SDA 和 SCL 上发出一个启动条件。
2. 在 SDA 和 SCL 上发出一个重复启动条件。
3. 写入 SSPBUF 寄存器，启动数据 / 地址的发送。
4. 配置 I^{2}C 端口用于接收数据。
5. 在接收数据字节末尾产生应答信号。
6. 在 SDA 和 SCL 上产生停止条件。

注：	当配置为 I^{2}C 主控模式时，MSSP 模块不允许事件排队。例如，在启动条件结束前，不允许用户立即写 SSPBUF 寄存器以启动传输。在这种情况下，将不会执行写 SSPBUF，WCOL 位将被置 1，这表明没有发生对 SSPBUF 的写操作。

下列事件会使 SSP 中断标志位 SSPIF 置 1 （如果使能 SSP 中断，则产生中断)：

- 启动条件
- 停止条件
- 数据字节发送 / 接收
- 应答发送
- 重复起始

图 15-16： MSSP 框图 （I^{2}C 主控模式）



15.4.6.1 I²C 主控模式工作方式

主器件产生所有串行时钟脉冲和启动 / 停止条件。以停止条件或重复启动条件结束传输过程。因为重复启动条件也是下一次串行传输的开始，因此 I²C 总线一直保持不被释放的状态。

在主控发送器模式下，串行数据通过 SDA 输出，而串行时钟由 SCL 输出。发送的第一个字节包括接收器件的从器件地址 （7 位）和读 / 写 （R/W）位。在这种情况下， R/W 位将是逻辑 “0”。一次发送 8 位串行数据。每发送一个字节，会收到一个应答位。输出启动和停止条件，表明串行传输的开始和结束。

在主控接收模式下，发送的第一个字节包括接收器件的地址 （7 位）和 R/W 位。在这种情况下， R/W 将是逻辑 “1”。因此，发送的第一个字节是一个 7 位从器件地址，后面跟 1 表示接收。串行数据通过 SDA 接收，而串行时钟由 SCL 输出。每次接收 8 位串行数据。每接收到一个字节，都会发送一个应答位。启动和停止条件分别代表发送的开始和结束。

在 I²C 模式下，在 SPI 模式工作中使用的波特率发生器用于将SCL时钟频率设置为100kHz、400kHz或1MHz。详情，请参见**第 15.4.7 节 “波特率”**。

下面是一个典型的发送时序：

1. 用户通过将起始使能位 SEN （SSPCON2<0>）置 1，产生启动条件。
2. SSPIF 置 1。在进行下一步操作前， MSSP 模块将等待所需的启动时间。
3. 用户将从器件地址装入 SSPBUF 进行发送。
4. 器件地址从 SDA 引脚移出，直到发送完所有 8 位地址数据。
5. MSSP 模块移入来自从器件的 ACK 位，并将它的值写入 SSPCON2 寄存器（SSPCON2<6>）。
6. MSSP 模块在第 9 个时钟周期的末尾将 SSPIF 置 1，产生一个中断。
7. 用户将 8 位数据装入 SSPBUF。
8. 数据从 SDA 引脚移出，直到发送完所有 8 位数据。
9. MSSP 模块移入来自从器件的 ACK 位，并将它的值写入 SSPCON2 寄存器（SSPCON2<6>）。
10. MSSP 模块在第 9 个时钟周期的末尾将 SSPIF 置 1，产生一个中断。
11. 用户通过将停止使能位 PEN （SSPCON2<2>）置 1 产生停止条件。
12. 一旦停止条件完成，将产生一个中断。

15.4.7 波特率

在 I²C 主控模式下，波特率发生器（Baud Rate Generator，BRG）的重载值位于 SSPADD 寄存器的低 7 位（图 15-17）。当发生对 SSPBUF 的写操作时，波特率发生器将自动开始计数。BRG 会递减计数至 0，然后停止直到再次发生重载。BRG计数器会在每个指令周期中的 Q2 和 Q4 时钟周期上（TCY）进行两次减计数。在 I²C 主控模式下，会自动重载 BRG。

如果指定操作完成（即，在传输的最后一个数据位后面跟着 ACK），内部时钟将自动停止计数，SCL 引脚将保持在其最后的状态。

表 15-3 给出了不同的指令周期下的时钟频率以及装入 SSPADD 的 BRG 值。

图 15-17： 波特率发生器框图

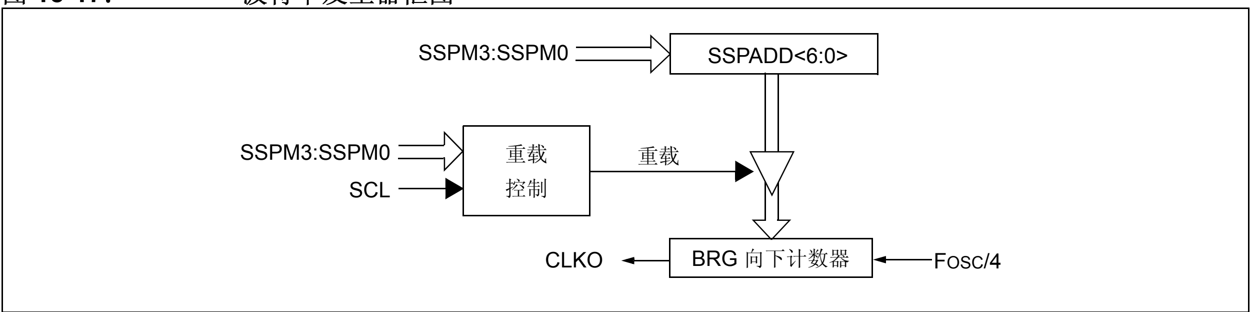


表 15-3： 使用 BRG 的 I²C 时钟频率

FCY	FCY*2	BRG 值	FSCL（两次 BRG 计满返回）
10 MHz	20 MHz	19h	400 kHz(1)
10 MHz	20 MHz	20h	312.5 kHz
10 MHz	20 MHz	3Fh	100 kHz
4 MHz	8 MHz	0Ah	400 kHz(1)
4 MHz	8 MHz	0Dh	308 kHz
4 MHz	8 MHz	28h	100 kHz
1 MHz	2 MHz	03h	333 kHz(1)
1 MHz	2 MHz	0Ah	100 kHz
1 MHz	2 MHz	00h	1 MHz(1)

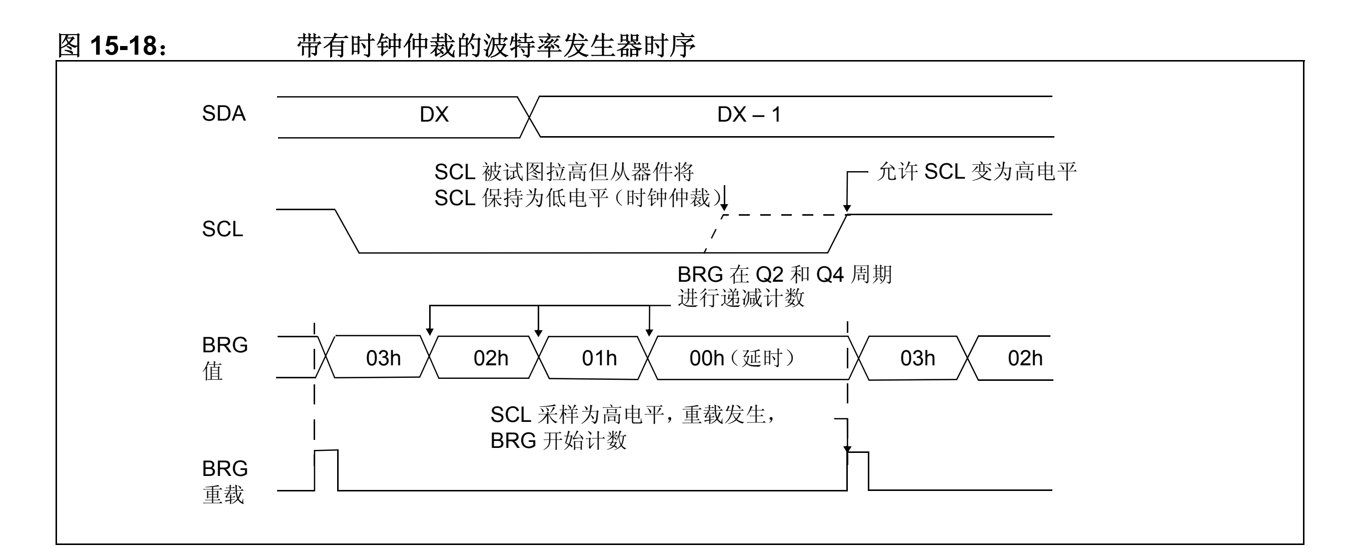
注 1：虽然 I²C 接口各方面都不符合 400 kHz I²C 规范（该规范适用于大于 100 kHz 的频率），但在需要较高频率的应用场合可以慎重使用。

 初稿

15.4.7.1 时钟仲裁

如果在任何接收、发送或重复启动 / 停止条件过程中，主器件拉高了 SCL 引脚（允许 SCL 引脚悬空为高电平），就会发生时钟仲裁。如果允许 SCL 引脚悬空为高电平，波特率发生器（BRG）将暂停计数直到实际采样到 SCL 引脚为高电平为止。当 SCL 引脚采样为高电平时，波特率发生器将被重新装入 SSPADD<6:0> 的值并开始计数。这可以保证当外部器件将时钟拉低时，SCL 在至少一个 BRG 周期内保持高电平（图 15-18）。

图 15-18： 带有时钟仲裁的波特率发生器时序



初稿

15.4.8 I²C 主控模式启动条件时序

要产生启动条件，用户应将启动条件使能位 SEN （SSPCON2<0>）置 1。当 SDA 和 SCL 引脚采样值为高电平时，波特率发生器重新装入 SSPADD<6:0> 的内容并开始计数。如果波特率发生器发生超时 （TBRG），SCL 和 SDA 都采样为高电平时，则 SDA 引脚被驱动为低电平。当 SCL 为高电平时，将 SDA 驱动为低电平将产生启动条件，并使 S 位 （SSPSTAT<3>）置 1。随后波特率发生器重新装入 SSPADD<6:0> 的值并恢复计数。当波特率发生器超时 （TBRG）时， SEN 位 （SSPCON2<0>）将自动被硬件清零，波特率发生器暂停工作， SDA 保持低电平，启动条件结束。

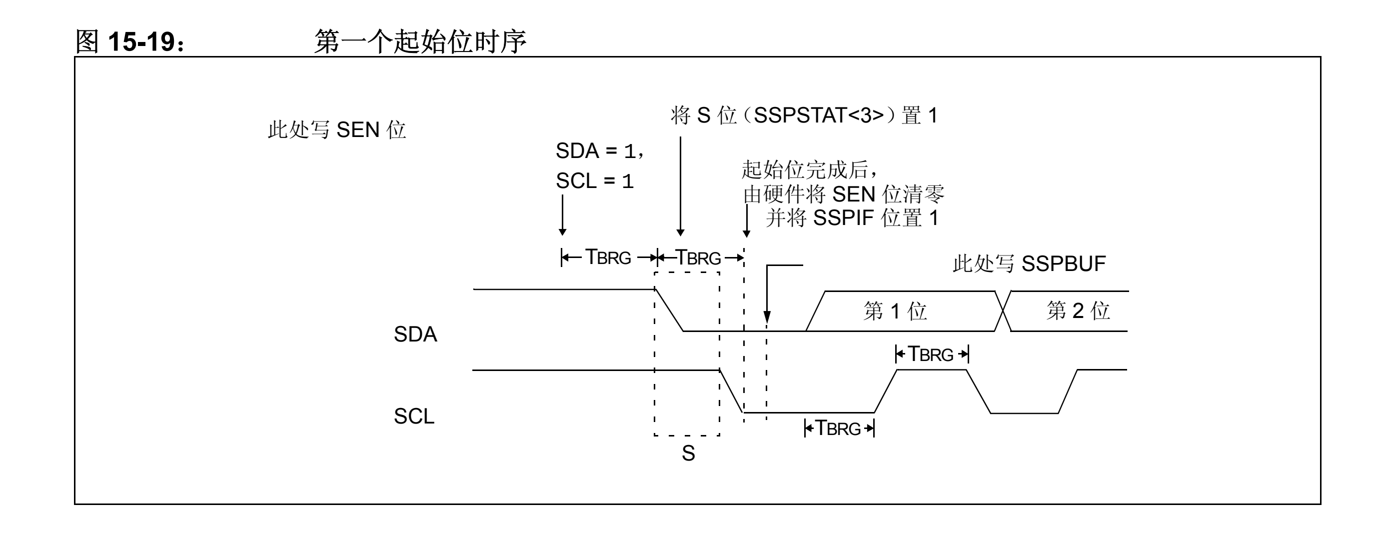
注：	如果在启动条件开始时，SDA 和 SCL 引脚已经采样为低电平，或者在启动条件期间，SCL 在 SDA 线被驱动为低电平之前已经采样为低电平，则会产生总线冲突，总线冲突中断标志位 BCLIF 置 1，启动条件中止，I²C 模块复位到空闲状态。

15.4.8.1 WCOL 状态标志

在起始时序进行当中，如果用户写 SSPBUF，则 WCOL 被置 1，同时缓冲器内容不变 （写操作无效）。

注：	由于不允许事件排队，在启动条件结束之前，不能对 SSPCON2 的低 5 位进行写操作。

图 15-19： 第一个起始位时序



15.4.9 I^2C 主控模式重复启动条件时序

将 RSEN 位（SSPCON2<1>）编程为高电平，并且 I^2C 逻辑模块处于空闲状态时，就会产生重复启动条件。当 RSEN 位置 1 时， SCL 引脚被拉为低电平。当 SCL 引脚采样为低电平时，波特率发生器装入 SSPADD<5:0> 的值，并开始计数。在一个波特率发生器计数周期（TBRG）后 SDA 引脚被释放 （其引脚电平被拉高）。当波特率发生器超时时，如果 SDA 采样为高电平，SCL 引脚将被拉高。当 SCL 被采样为高电平时，波特率发生器重新装入 SSPADD<6:0> 的值并开始计数。 SDA 和 SCL 必须在一个计数周期 TBRG 内采样为高电平。接下来，当 SCL 为高电平时，在一个 TBRG 中，将 SDA 引脚驱动为（SDA = 0）低电平。然后 RSEN 位（SSPCON2<1>）将自动清零，波特率发生器不会重载，SDA 引脚保持低电平。一旦在 SDA 和 SCL 引脚上检测到启动条件， S 位（SSPSTAT<3>）将被置 1。直到波特率发生器发生超时后， SSPIF 位才会置 1。

> **注 1：** 有其他事件在进行时，编程设置对 RSEN 无效。
>
> **2：** 在重复启动条件期间，下列事件将会导致总线冲突：
>
> • 当 SCL 由低电平变为高电平时， SDA 采样为低电平。
>
> • 在 SDA 被拉低之前， SCL 变为低电平。这表明另一个主器件正试图发送一个数据 1。

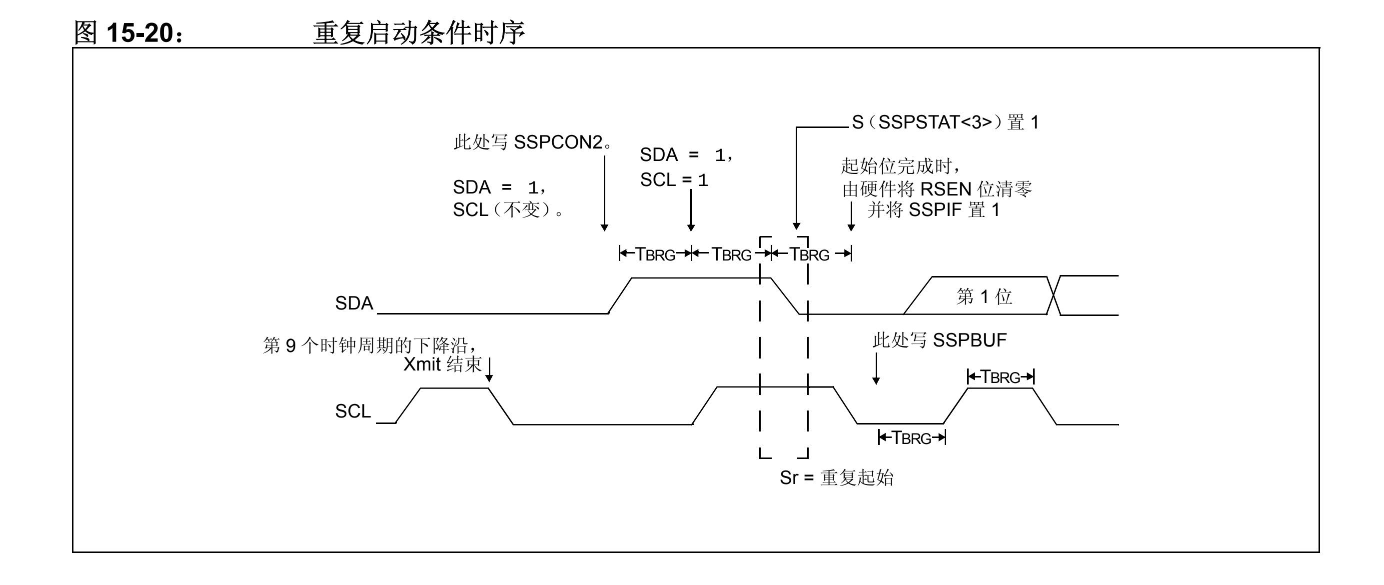
一旦 SSPIF 位被置 1，用户便可以在 7 位地址模式下将 7 位地址，或者在 10 位地址模式下将默认的第一个地址字节写入 SSPBUF。当发送完第一个 8 位数据并接收到一个 ACK 后，用户可以发送另外 8 位地址 （10 位地址模式）或 8 位数据 （7 位地址模式）。

15.4.9.1 WCOL 状态标志

在重复起始时序进行当中，如果用户写 SSPBUF，则 WCOL 被置 1，同时缓冲器内容不变（写操作无效）。

> **注：** 由于不允许事件排队，在重复启动条件结束之前，不能对 SSPCON2 的低 5 位进行写操作。

图 15-20： 重复启动条件时序



 初稿

15.4.10 I²C 主控模式下的发送

发送一个数据字节、一个 7 位地址或一个 10 位地址的另一半，都是通过写一个值到 SSPBUF 寄存器来实现的。该操作将使缓冲器满标志位 BF 置 1，波特率发生器开始计数，同时开始下一次发送。在 SCL 的下降沿有效后（见数据保持时间规范参数 #106），地址 / 数据的每一位被移出至 SDA 引脚。在一个波特率发生器计满返回周期（TBRG）内，SCL 保持低电平。数据应该在 SCL 释放为高电平前保持有效（见数据建立时间规范参数 #107）。当 SCL 引脚释放为高电平时，它将在 TBRG 中保持高电平状态。在此期间以及 SCL 的下一个下降沿之后的一段时间内，SDA 引脚上的数据必须保持稳定。在第 8 位数据被移出（第 8 个时钟周期的下降沿）之后，BF 标志位被清零，同时主器件释放 SDA。此时如果发生地址匹配或是数据被正确接收，被寻址的从器件将在第 9 个时钟周期发出一个 ACK 位响应。ACK 的状态在第 9 个时钟周期的下降沿写入 ACKDT 位。主器件接收到应答之后，应答状态位 ACKSTAT 会被清零。如果未收到应答，则该位被置 1。第 9 个时钟周期之后，SSPIF 位会置 1，主控时钟（波特率发生器）暂停，直到下一个数据字节装入 SSPBUF，SCL 引脚保持低电平，SDA 保持不变（图 15-21）。

在写 SSPBUF 之后，地址的每一位在 SCL 的下降沿被移出，直至所有 7 位地址位和 R/W 位都被移出。在第 8 个时钟的下降沿，主器件将 SDA 引脚拉为高电平，以允许从器件发出一个应答响应。在第 9 个时钟的下降沿，主器件通过采样 SDA 引脚来判断地址是否被从器件识别。ACK 位的状态被装入 ACKSTAT 状态位（SSPCON2<6>）。在发送地址的第 9 个时钟下降沿之后，SSPIF 置 1，BF 标志位清零，波特率发生器关闭直到下一次写 SSPBUF，且 SCL 引脚保持低电平，允许 SDA 引脚悬空。

15.4.10.1 BF 状态标志

在发送模式下，BF 位（SSPSTAT<0>）在 CPU 写 SSPBUF 时置 1，在所有 8 位数据移出后清零。

15.4.10.2 WCOL 状态标志

如果用户在发送过程中（即，SSPSR 仍在移出数据字节时）写 SSPBUF，则 WCOL 置 1，缓冲器内容不变（写操作无效）。

WCOL 必须用软件清零。

15.4.10.3 ACKSTAT 状态标志

在发送模式下，当从器件发送应答响应（ACK = 0）时，ACKSTAT 位（SSPCON2<6>）清零；当从器件没有应答（ACK = 1）时，该位置 1。从器件在识别出其地址（包括广播呼叫地址）或正确接收数据后，会发出一个应答。

15.4.11 I²C 主控模式接收

通过编程接收使能位 RCEN（SSPCON2<3>）使能主控模式接收。

注：	RCEN 位置 1 前，MSSP 必须处于空闲状态，否则上述操作无效。

波特率发生器开始计数，每次计满返回时，SCL 引脚的状态发生改变（由高变低或由低变高），数据被移入 SSPSR。第 8 个时钟的下降沿之后，接收使能标志位自动清零，SSPSR 的内容装入 SSPBUF，BF 标志位置 1，SSPIF 标志位置 1，波特率发生器暂停计数，且 SCL 保持为低电平。此时 MSSP 处于空闲状态，等待下一条命令。当 CPU 读缓冲器时，BF 标志位将自动清零。通过将应答时序使能位 ACKEN（SSPCON2<4>）置 1，用户可以在接收结束后发送应答位。

15.4.11.1 BF 状态标志

接收数据过程中，把地址或数据字节从 SSPSR 装入 SSPBUF 时，BF 位置 1。在读 SSPBUF 寄存器时将其清零。

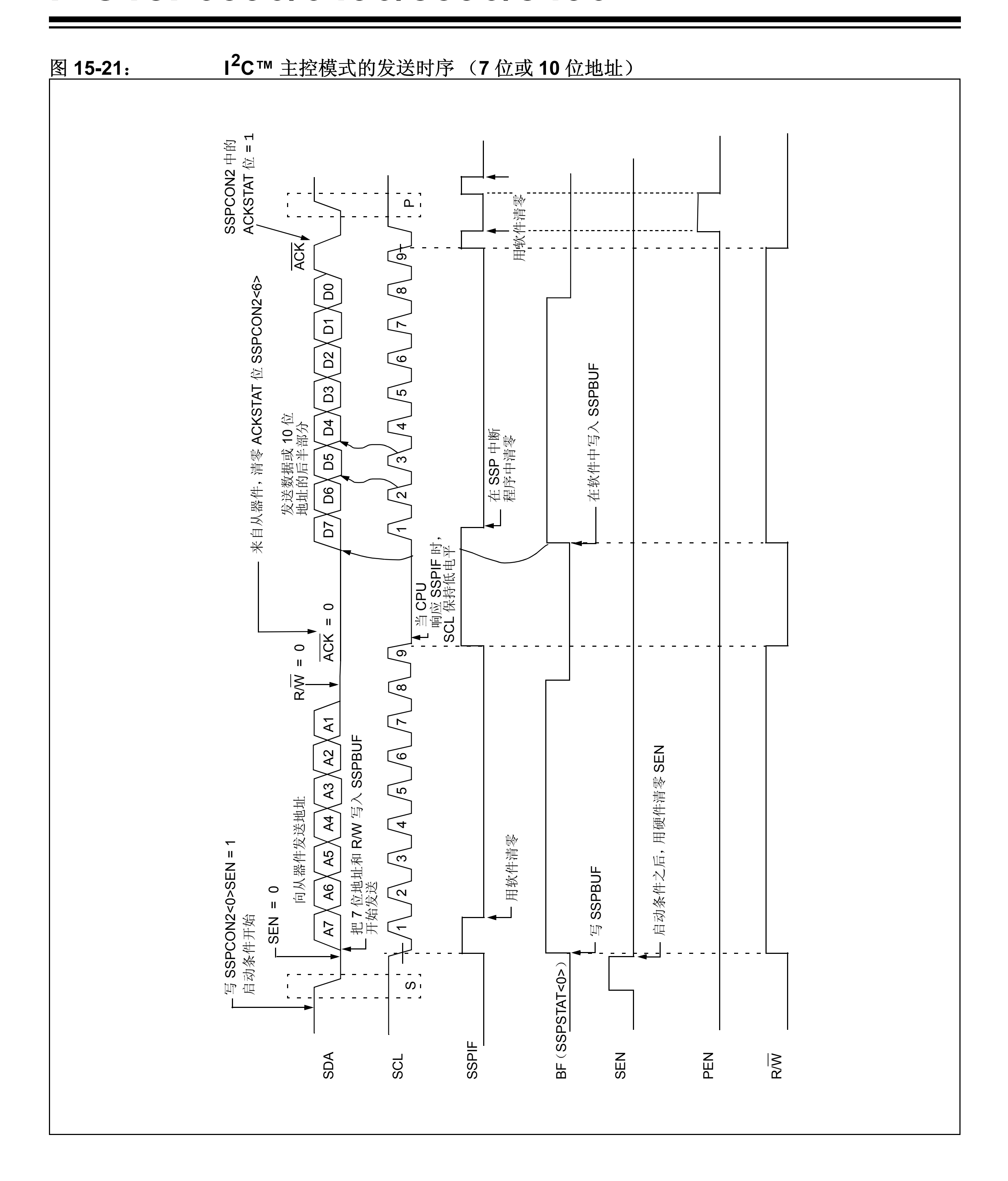
15.4.11.2 SSPOV 状态标志

接收数据过程中，当SSPSR接收到8位数据时，SSPOV 位置 1，BF 标志位已经在上一次接收时置 1。

15.4.11.3 WCOL 状态标志

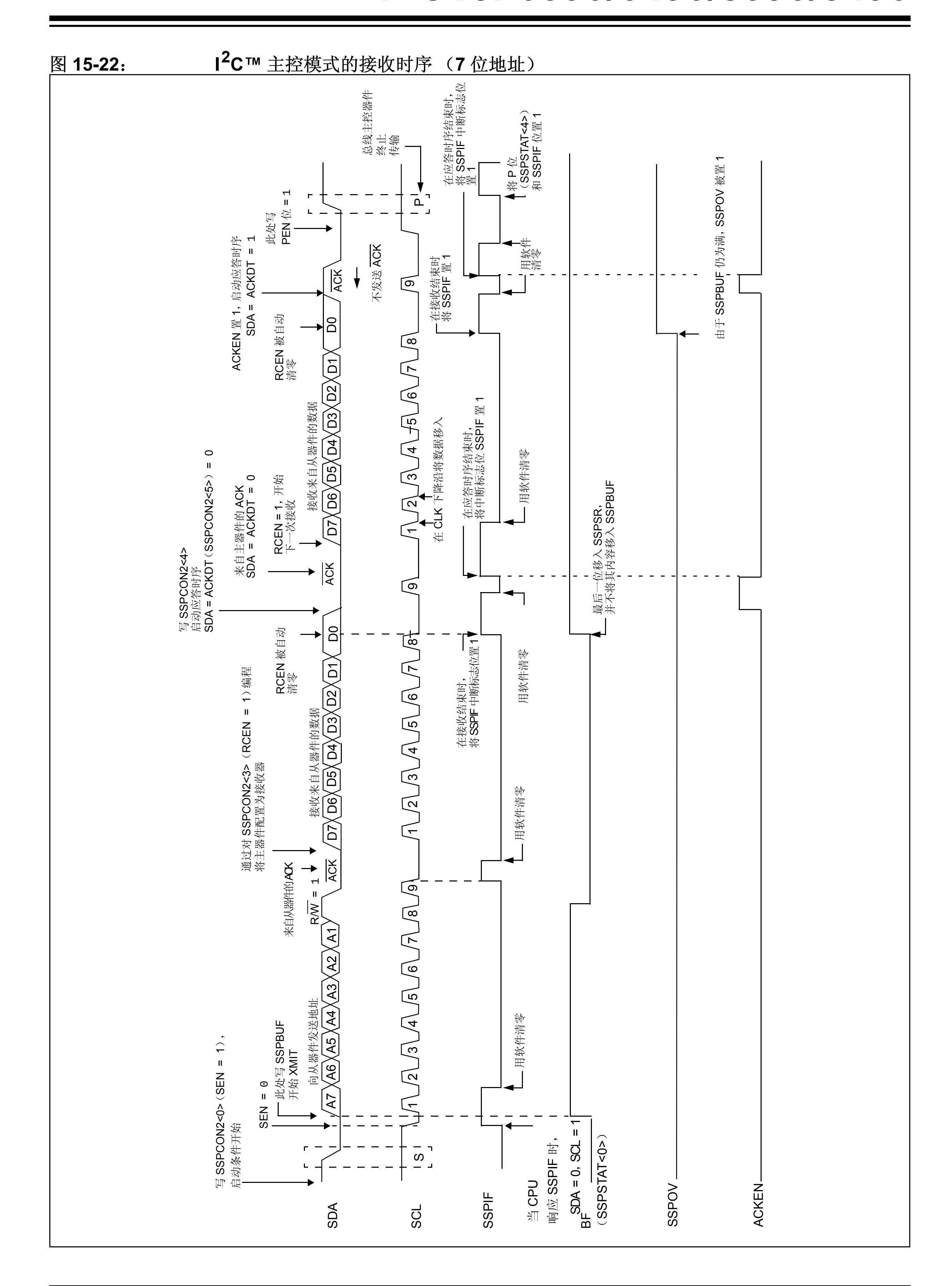
如果用户在接收过程中（即，SSPSR 仍在移入数据字节时）写 SSPBUF，则 WCOL 位置 1，缓冲器内容不变（写操作无效）。

图 15-21： I^2C™ 主控模式的发送时序 （7 位或 10 位地址）



 初稿

图 15-22：　　　　I²C™ 主控模式的接收时序 （7 位地址）



15.4.12 应答序列时序

将应答序列使能位 ACKEN（SSPCON2<4>）置 1 即可使能应答序列。当该位被置 1 时， SCL 引脚被拉低，应答数据位的内容输出到 SDA 引脚上。如果用户希望产生一个应答，则应该将 ACKDT 位清零。否则，用户要在应答序列开始前将 ACKDT 位置 1。然后波特率发生器进行一个计满返回周期 （TBRG）的计数， SCL 引脚电平被拉高。当 SCL 引脚采样为高电平时 （时钟仲裁），波特率发生器再进行一个 TBRG 周期的计数。然后 SCL 引脚被拉低。在这之后，ACKEN 位自动清零，波特率发生器关闭， MSSP 模块进入空闲模式 （图 15-23）。

15.4.12.1 WCOL 状态标志

如果用户在应答序列进行过程中试图写 SSPBUF，则 WCOL 将置 1，缓冲器的内容不会改变（写操作无效）。

15.4.13 停止条件时序

将停止时序使能位 PEN （SSPCON2<2>）置 1，在接收 / 发送结束后， SDA 引脚上将产生停止位。在接收 / 发送结束时，SCL 引脚在第 9 个时钟的下降沿后保持低电平。当 PEN 位置 1 时，主器件将 SDA 线置为低电平。当 SDA 线采样为低电平时，波特率发生器被重载并递减计数至 0。当波特率发生器发生超时，SCL 引脚被拉为高电平，在一个 TBRG （波特率发生器计满返回周期）之后， SDA 引脚将被拉高。当 SDA 引脚采样为高电平且 SCL 也是高电平时， P 位 （SSPSTAT<4>）置 1。另一个 TBRG 之后，PEN 位被清零，同时 SSPIF 位被置 1 （图 15-24）。

15.4.13.1 WCOL 状态标志

如果用户在停止时序过程中试图写SSPBUF，则WCOL 位将置 1，缓冲器的内容不会改变 （写操作无效）。

图 15-23： 应答序列波形

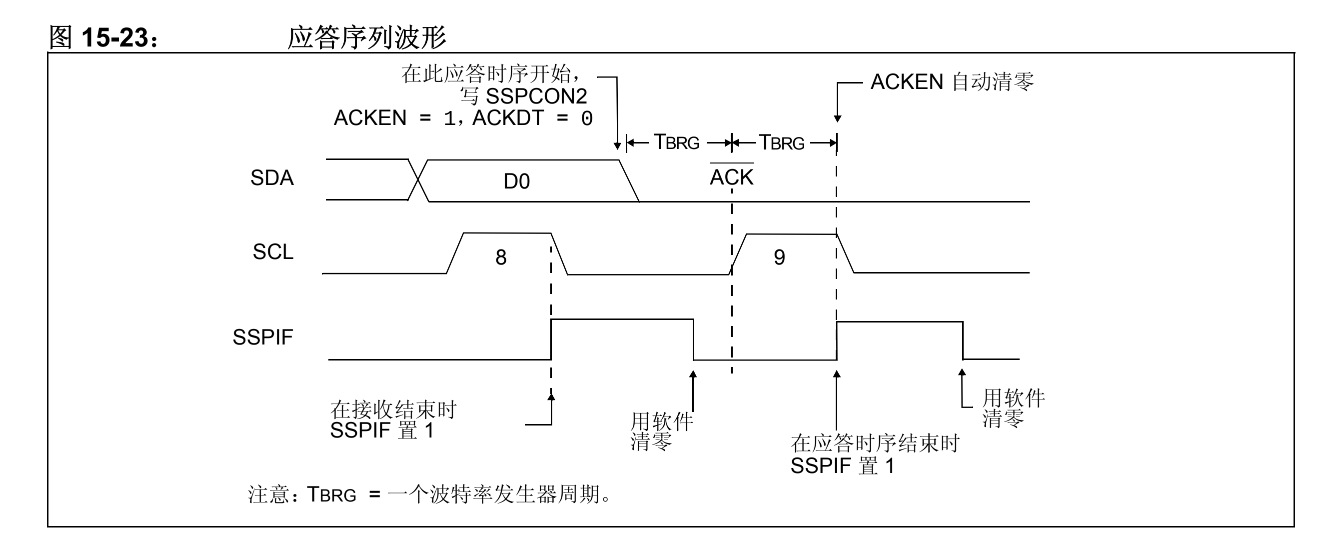
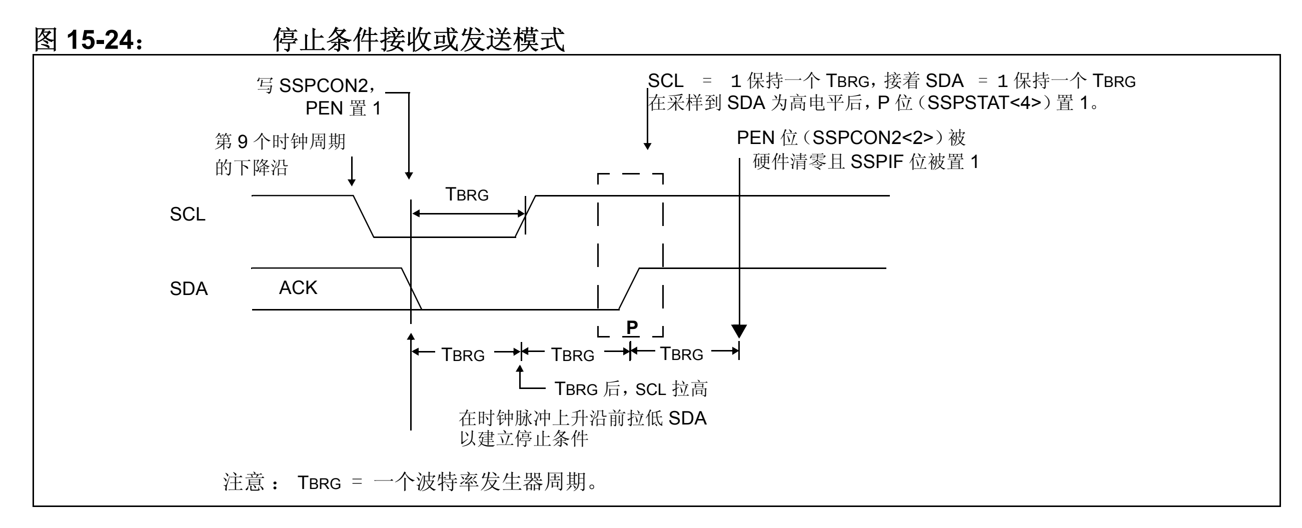


图 15-24： 停止条件接收或发送模式



15.4.14 休眠工作方式

在休眠模式下， I²C 模块能够接收地址或数据。并且在地址匹配或字节传输完成后，如果使能 MSSP 中断，将唤醒处理器。

15.4.15 复位效果

复位操作会禁止 MSSP 模块并终止当前的传输 。

15.4.16 多主机模式

在多主机模式下，在检测到启动和停止条件时将产生中断，这可以用于判断总线是否空闲。停止 （P）位和启动 （S）位在复位时或禁止 MSSP 模块时被清零。当 P 位 （SSPSTAT<4>）置 1，可以取得 I²C 总线的控制权 ；否则， P 位和 S 位都清零，总线处于空闲状态。当总线忙时，一旦出现停止条件，将产生中断。

在多主机模式下，必须一直监视 SDA 线，查看信号电平是否为期望的输出电平。此操作由硬件实现，其结果保存在 BCLIF 位中。

可能导致仲裁失败的情况是：

- 地址传输
- 数据传输
- 启动条件
- 重复的启动条件
- 应答信号

15.4.17 多主机通信、总线冲突与总线仲裁

多主机模式是通过总线仲裁来支持的。当主器件将地址/数据位输出到SDA引脚时，如果一个主器件在SDA上输出1，而另一个主器件输出0，就会发生总线仲裁。如果SDA引脚上期望的数据是1，而实际采样到的数据是0，则发生了总线冲突。主器件将把总线冲突中断标志位BCLIF 置1，并将I²C端口复位到空闲状态（图 15-25）。如果在发送过程中发生总线冲突，则发送操作停止，BF标志位被清零，SDA和SCL线被拉高，并且将SSPBUF置于可写入状态。当执行完总线冲突中断服务程序后，如果I²C总线空闲，用户可通过发出启动条件恢复通信。

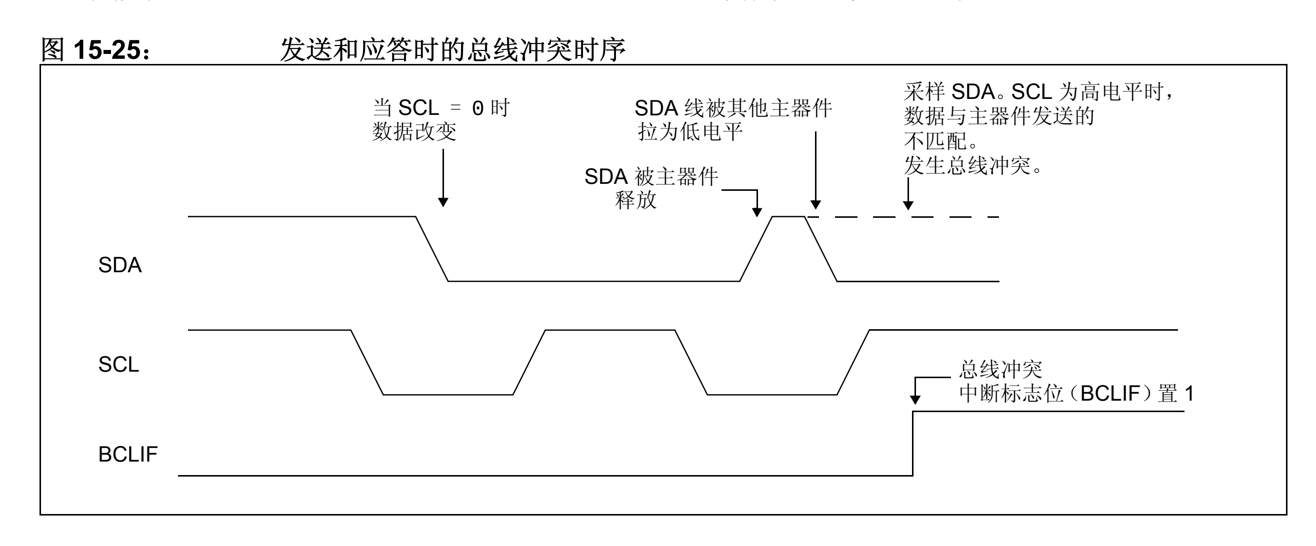
如果在起始、重复起始、停止或应答信号的执行过程中发生总线冲突，则这种状态被中止，SDA 和 SCL 线被拉高， SSPCON2 寄存器中的对应控制位清零。当执行完总线冲突中断服务程序后，如果 I²C 总线空闲，用户可通过发出启动条件恢复通信。

主器件将继续监视 SDA 和 SCL 引脚。一旦出现停止条件， SSPIF 位将被置 1。

发生总线冲突时无论发送的进度如何，写入SSPBUF 都会从第一个数据位开始发送数据。

在多主机模式下，通过在检测到起始和停止条件时产生中断可以确定总线何时空闲。 SSPSTAT 寄存器中的 P 位置 1，可以获取 I²C 总线的控制权；否则， P 位和 S 位都清零，总线处于空闲状态。

图 15-25： 发送和应答时的总线冲突时序



15.4.17.1 启动条件期间的总线冲突

启动条件期间，以下事件将导致总线冲突：

a） 在启动条件开始时， SDA 或 SCL 被采样为低电平 （图 15-26）。

b） SDA被拉低之前，SCL采样为低电平（图 15-27）。

在启动条件期间， SDA 和 SCL 引脚都会被监视。

如果 SDA 引脚已经是低电平，或 SCL 引脚已经是低电平，则：

- 中止启动条件，
- BCLIF 标志位置 1，
- MSSP 模块复位为空闲状态 （图 15-26）。

启动条件从 SDA 和 SCL 引脚被拉高开始。当 SDA 引脚采样为高电平时，波特率发生器装入 SSPADD<6:0> 的值并递减计数至 0。如果在 SDA 为高电平时，SCL 引脚采样为低电平，则发生总线冲突，因为这表示另一台主器件在启动条件期间试图发送一个数据 “1”。

如果 SDA 引脚在该计数周期内采样为低电平，则 BRG 复位，同时 SDA 线保持原值（图 15-28）。但是，如果 SDA 引脚采样为 “1”， SDA 引脚将在 BRG 计数结束时被置为低电平。接着，波特率发生器被重载并递减计数至 0，在此期间，如果 SCL 引脚采样到 “0”，则不会发生总线冲突。在 BRG 计数结束时， SCL 引脚被拉为低电平。

注：	在启动条件期间不太可能发生总线冲突，因为两个总线主控器件不可能精确地在同一时刻发出启动条件。因此一个主器件将总是先于另一个主器件将 SDA 拉低。但是上述情况不会引起总线冲突，因为必须允许两个主器件对启动条件后的第一个地址进行仲裁。如果地址是相同的，必须继续对数据部分、重复启动条件或停止条件进行仲裁。

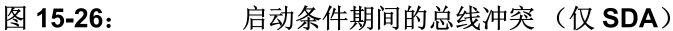
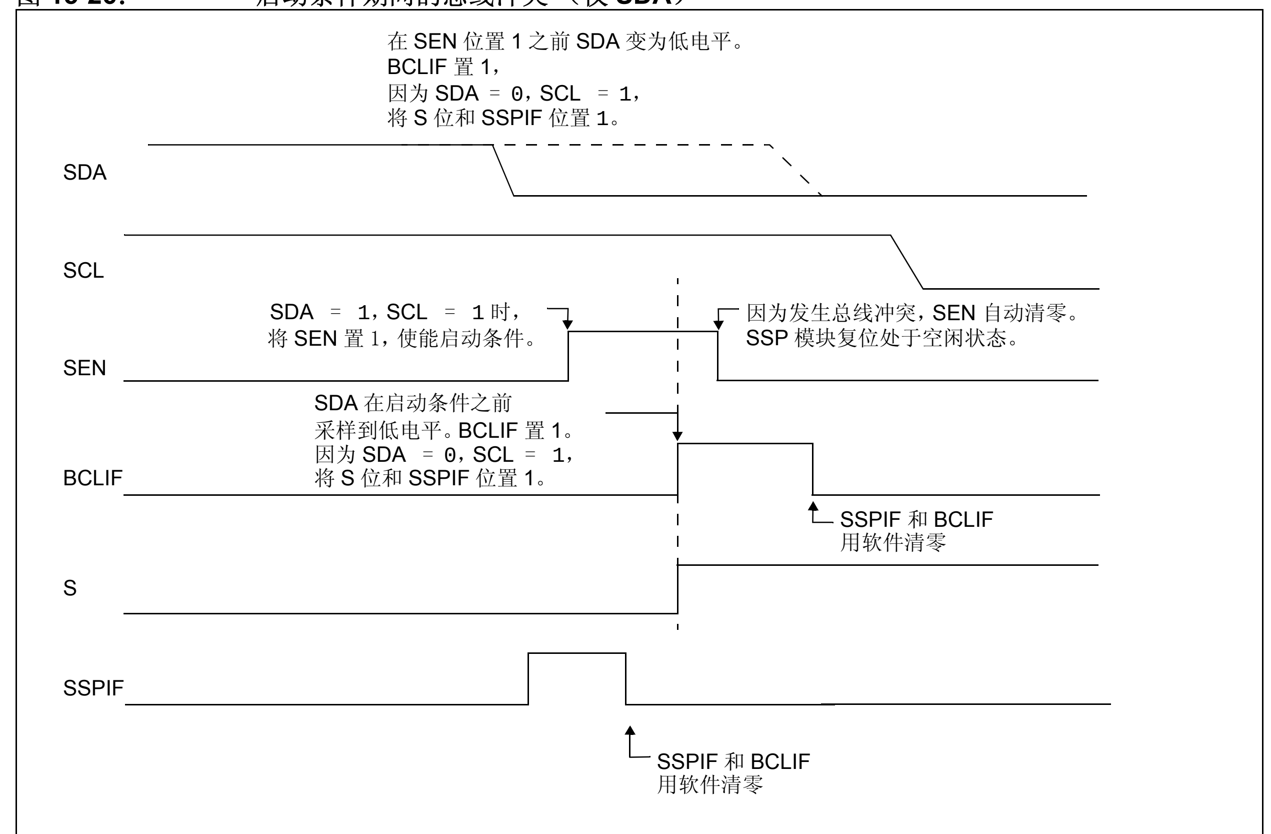


图 15-26： 启动条件期间的总线冲突 （仅 SDA）



 初稿

图 15-27： 启动条件期间的总线冲突 （SCL = 0）

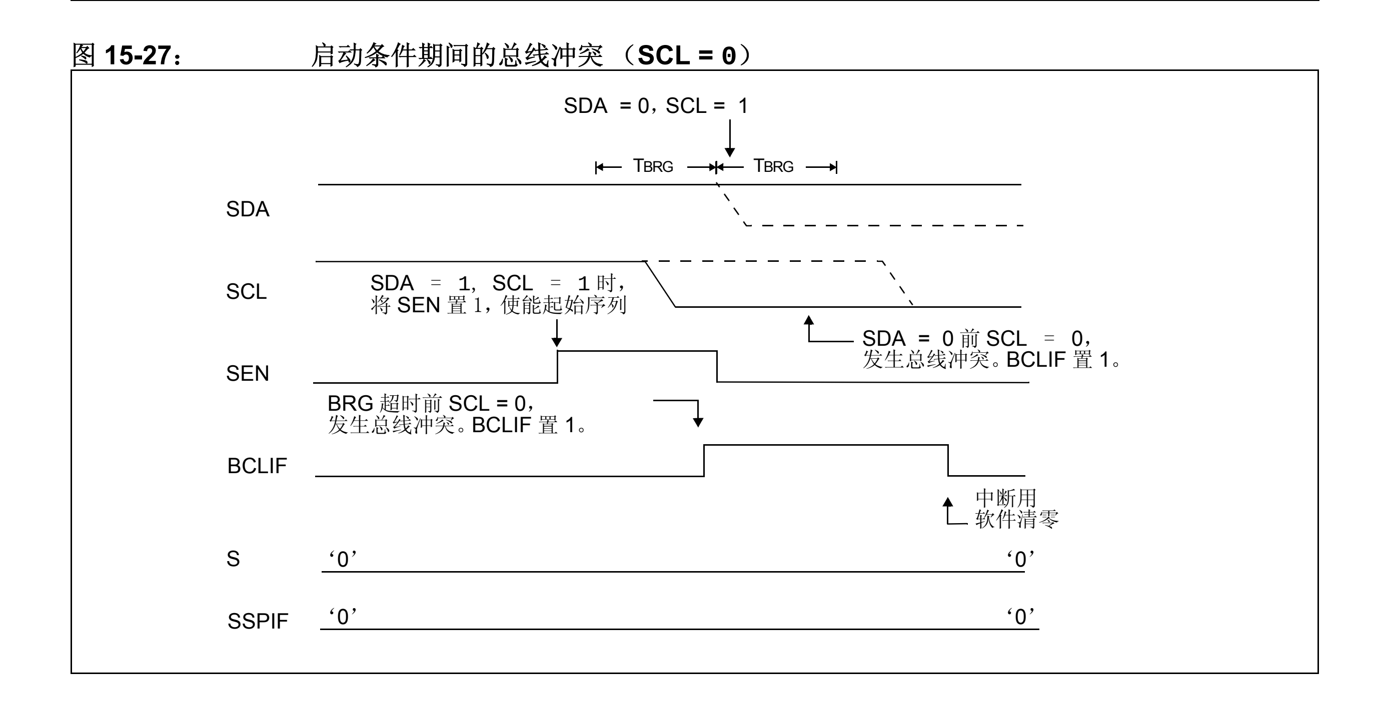
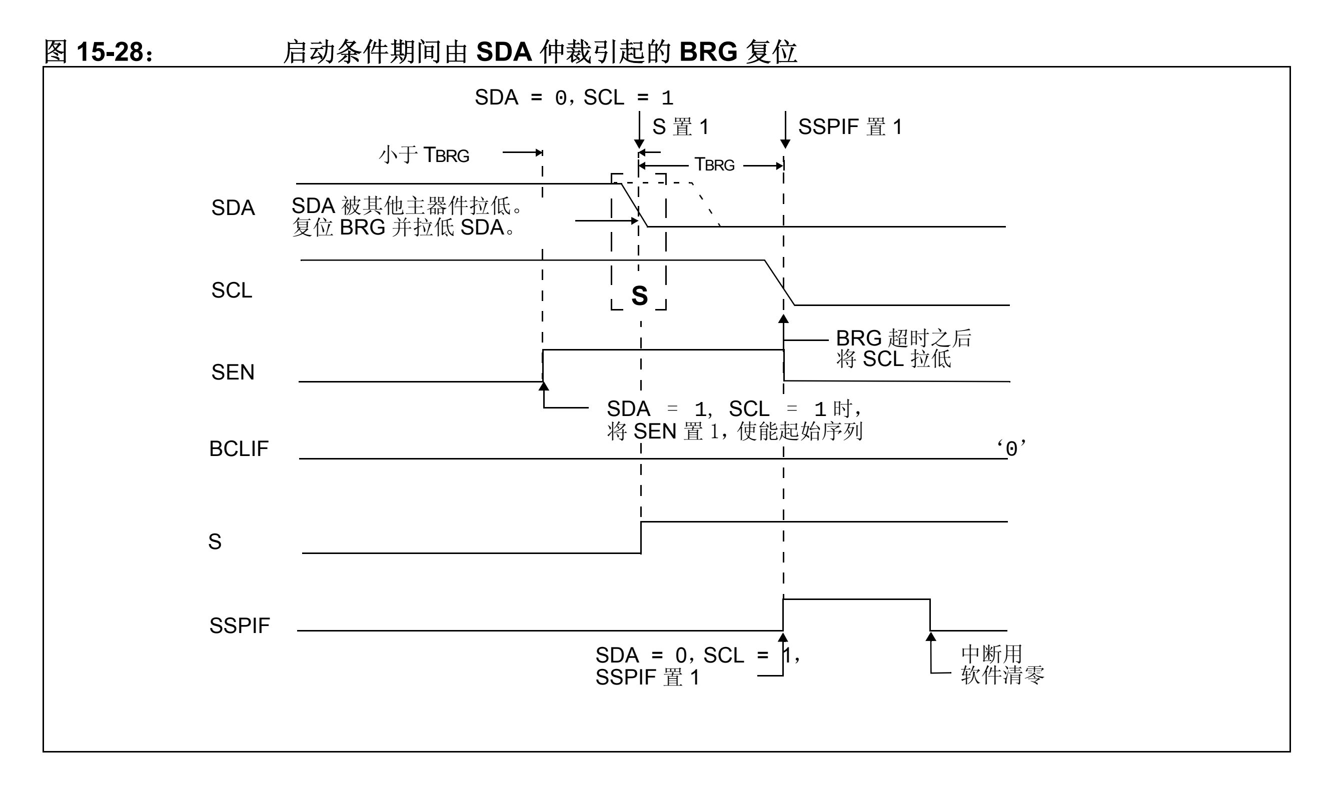


图 15-28： 启动条件期间由 SDA 仲裁引起的 BRG 复位



15.4.17.2 重复启动条件期间的总线冲突

在下列情况中，重复启动条件期间会发生总线冲突：

a） 在 SCL 由低电平变为高电平期间，在 SDA 上采样到低电平。

b) 在 SDA 被拉为低电平之前， SCL 变为低电平，表示另一个主器件正试图发送一个数据 “1”。

当用户拉高 SDA 并允许该引脚悬空时， BRG 装入 SSPADD<6:0> 中的值并递减计数至 0，接着 SCL 引脚被置为高电平，当 SCL 引脚采样到高电平时，对 SDA 引脚进行采样。

如果 SDA 为低电平，则已发生了总线冲突 （即，另一个主器件正试图发送一个数据 “0”，图 15-29)。如果 SDA 被采样到高电平，则 BRG 被重新装入值并开始计数。如果 SDA 在 BRG 超时之前从高电平变为低电平，则不会发生总线冲突，因为两个主器件不可能精确地在同一时刻将 SDA 拉低。

如果 SCL 在 BRG 超时之前从高电平变为低电平，且 SDA 尚未被拉低，那么将发生总线冲突。在此情况下，另一个主控器在重复启动条件期间正试图发送一个数据 1 （见图 15-30）。

如果在BRG计时结束时SCL和SDA都仍然是高电平，则 SDA 引脚被拉低， BRG 重新装入值并开始计数。在计数结束时，不管 SCL 引脚的状态如何， SCL 引脚都被拉低，重复启动条件结束。

图 15-29： 重复启动条件期间的总线冲突 （情形 1）

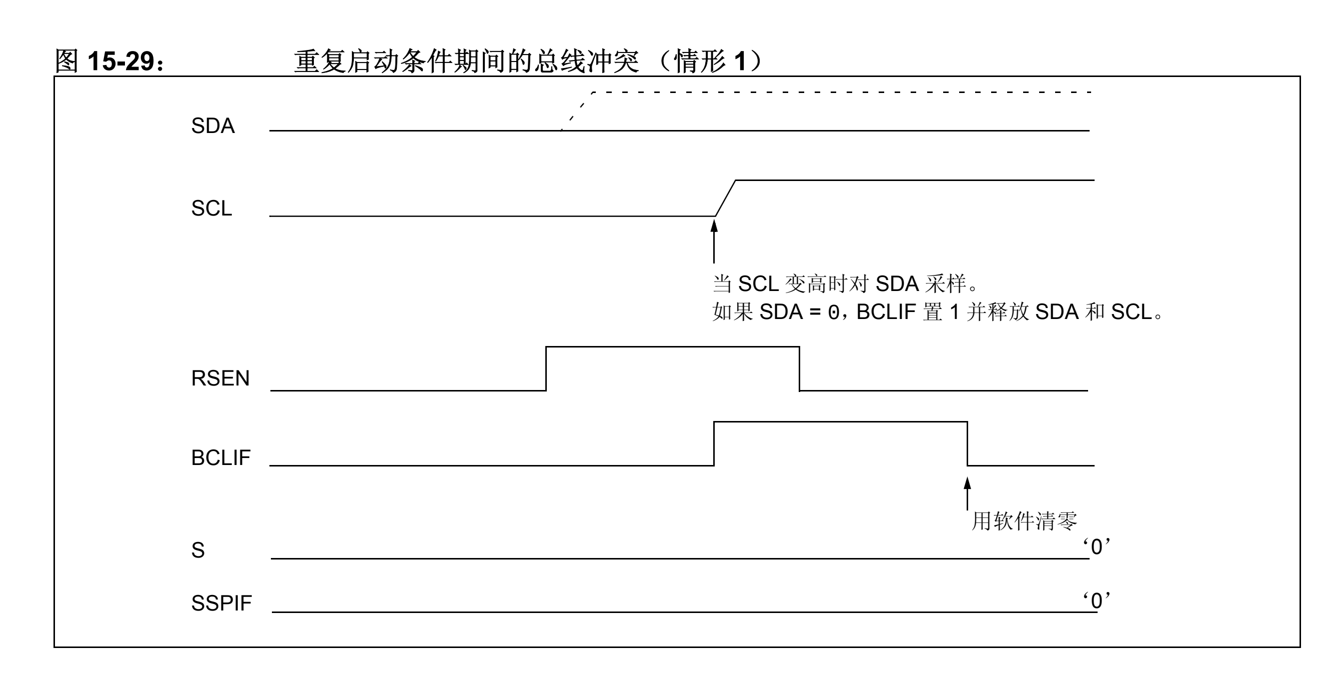
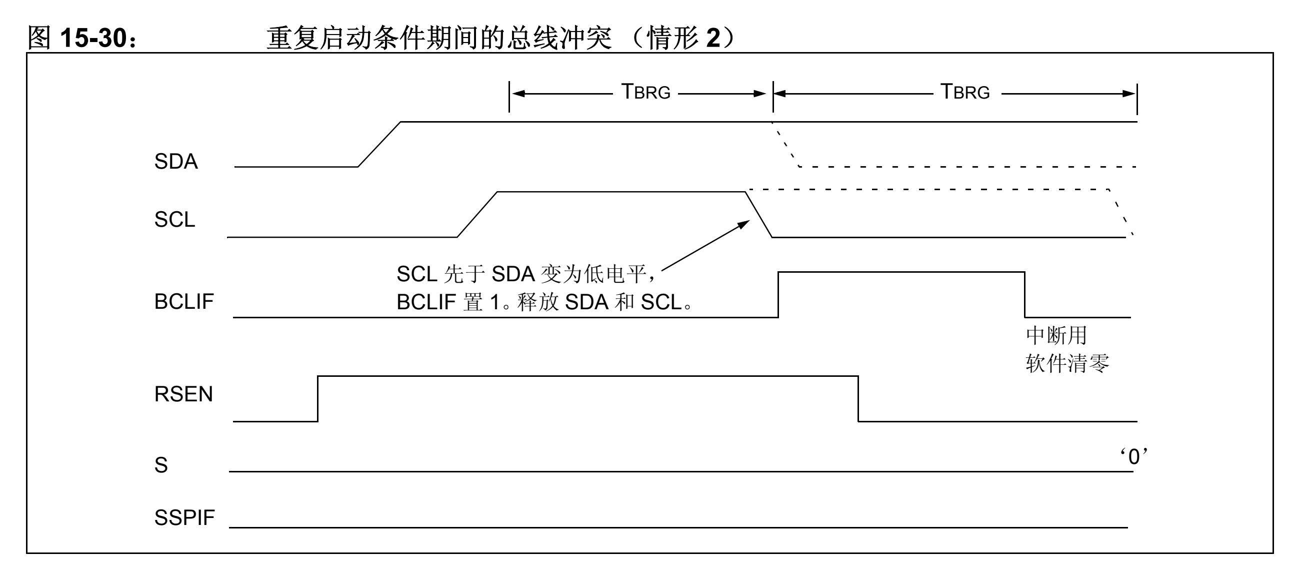


图 15-30： 重复启动条件期间的总线冲突 （情形 2）



 初稿

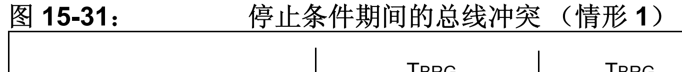
15.4.17.3 停止条件期间的总线冲突

以下事件会导致停止条件期间发生总线冲突：

a） SDA已被拉高并允许悬空为高电平之后，SDA在BRG 超时后被采样到低电平。

b) SCL 引脚被拉高之后，SCL 在 SDA 变成高电平之前被采样到低电平。

停止条件从 SDA 被置成低电平开始。当 SDA 采样为低电平时， SCL 引脚被允许悬空。当 SDA 被采样到高电平时 （时钟仲裁），波特率发生器装入 SSPADD<6:0> 的值并递减计数到 0。 BRG 超时后， SDA 被采样。如果 SDA 采样为低电平，则已发生总线冲突。这是因为另一个主器件正试图发送一个数据 “0”（图 15-31）。如果 SCL 引脚在允许 SDA 悬空为高电平前被采样到低电平，也会发生总线冲突。原因同上所述 （情形 2）（图 15-32）。


图 15-31： 停止条件期间的总线冲突 （情形 1）

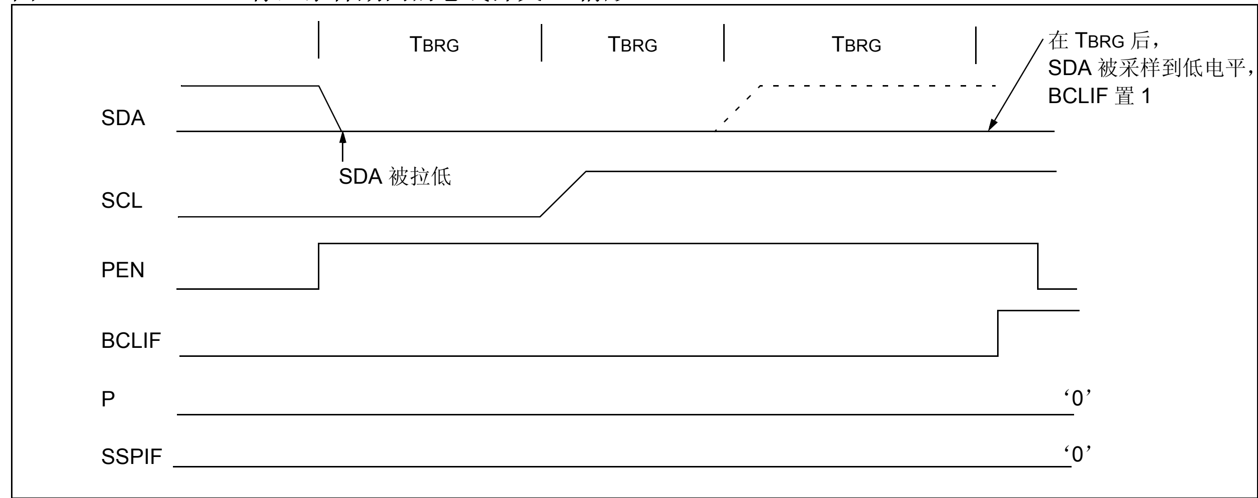



图 15-32： 停止条件期间的总线冲突 （情形 2）

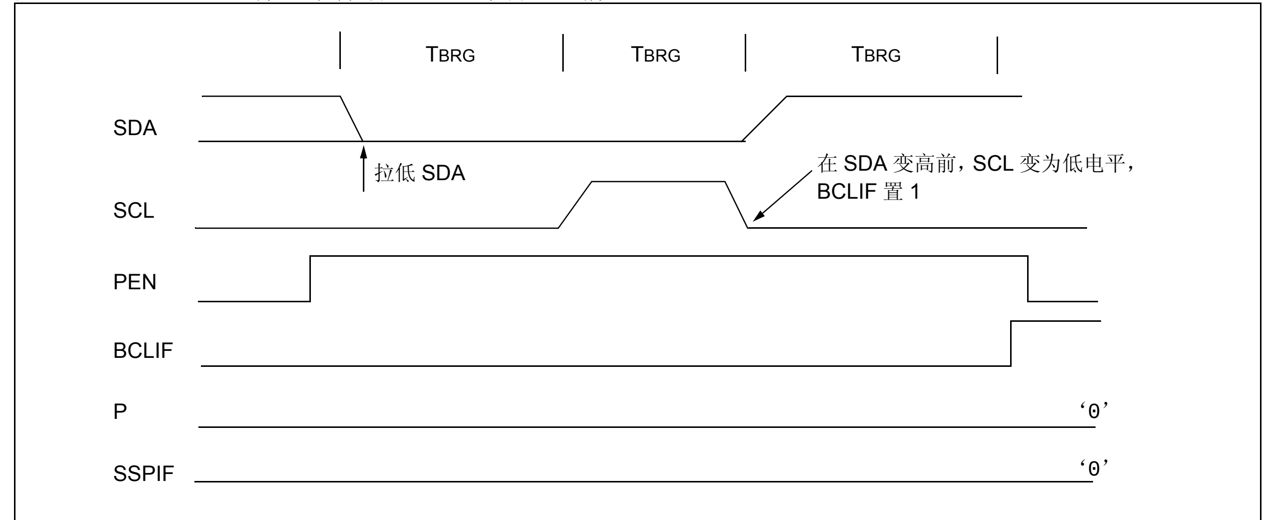


表 **15-4：** 与 **I²C** 工作模式相关的寄存器

名称	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
PIR1	—	ADIF	RC1IF	TX1IF	SSPIF	CCP1IF	TMR2IF	TMR1IF	61
PIE1	—	ADIE	RC1IE	TX1IE	SSPIE	CCP1IE	TMR2IE	TMR1IE	61
IPR1	—	ADIP	RC1IP	TX1IP	SSPIP	CCP1IP	TMR2IP	TMR1IP	61
TRISC	PORTC 数据方向寄存器								62
SSPBUF	同步串口接收缓冲器 / 发送寄存器								60
SSPADD	同步串口接收缓冲器 / 发送寄存器								60
SSPCON1	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	60
SSPCON2	GCEN	ACKSTAT	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN	60
SSPSTAT	SMP	CKE	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF	60

图注： — = 未用，读为 0。 SPI 模式下的 MSSP 未使用阴影单元。

 初稿

16.0 增强型通用同步 / 异步收发器（EUSART）

PIC18F6390/6490/8390/8490 器件有三个串行 I/O 模块：一个是在上一章中讨论过的 MSSP 模块，另外还有两个通用同步 / 异步收发器（Universal Synchronous Asynchronous Receiver Transmitter，USART）模块。通常，也将 USART 称为串行通信接口或 SCI。USART 可以被配置为能与 CRT 终端和个人计算机等外设通信的全双工异步系统；也可以被配置成能够与 A/D 或 D/A 集成电路、串行 EEPROM 等外设通信的半双工同步系统。

USART 模块在这些器件中有着两种完全不同的实现方式：一种是这里讨论的增强型 USART（EUSART），另一种是下一章中要讨论的可寻址 USART。对于该系列器件，USART1 指的是 EUSART，而 USART2 指的是 AUSART。

EUSART 和 AUSART 模块在串行通讯中实现了相同的核心功能，它们的基本操作也大致相同。EUSART 模块提供了更多的功能，包括自动波特率检测和校准，以及在接收到"同步中断"字符和发送 12 位间隔字符时自动唤醒。这些功能使 EUSART 模块成为局域互连网络（Local Interconnect Network，LIN）总线系统理想的选择。

可将 EUSART 配置为以下几种工作模式：

- 带有以下功能的全双工异步模式：
  - 字符接收自动唤醒
  - 自动波特率校准
  - 12 位间隔字符发送
- 半双工同步主控模式（时钟极性可选）
- 半双工同步从动模式（时钟极性可选）

EUSART 的引脚与 PORTC（RC6/TX1/CK1 和 RC7/RX1/DT1）的功能复用。要把这些引脚配置为 EUSART：

- SPEN（RCSTA1<7>）位必须置 1
- TRISC<7> 位必须置 1
- TRISC<6> 位必须置 1

注：	USART控制在需要时会自动将引脚从输入重新配置为输出。

增强型 USART 模块的操作是由 3 个寄存器控制的：

- 发送状态和控制寄存器 1（TXSTA1）
- 接收状态和控制寄存器 1（RCSTA1）
- 波特率控制寄存器 1（BAUDCON1）

这些寄存器将在寄存器 16-1、寄存器 16-2 和寄存器 16-3 中介绍。

寄存器 16-1： TXSTA1：EUSART 发送状态和控制寄存器

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-1	R/W-0
CSRC	TX9	TXEN(1)	SYNC	SENDB	BRGH	TRMT	TX9D
bit 7							bit 0

bit 7 **CSRC**：时钟源选择位

异步模式：

忽略。

同步模式：

1 = 主控模式 （时钟来自内部 BRG）

0 = 从动模式 （时钟来自外部时钟源）

bit 6 **TX9**：9 位发送使能位

1 = 选择 9 位发送

0 = 选择 8 位发送

bit 5 **TXEN**：发送使能位 (1)

1 = 使能发送

0 = 禁止发送

注 1：同步模式下 SREN/CREN 的优先级高于 TXEN。

bit 4 **SYNC**：AUSART 模式选择位

1 = 同步模式

0 = 异步模式

bit 3 **SENDB**：发送间隔字符位

异步模式：

1 = 在下一次发送时发送 “同步中断” 字符 （在完成时由硬件清零）

0 =“同步中断”字符发送完成

同步模式：

忽略。

bit2 **BRGH**：高波特率选择位

异步模式：

1 = 高速

0 = 低速

同步模式：

在此模式下未使用。

bit1 **TRMT**：发送移位寄存器状态位

1 = TSR 空

0 = TSR 满

bit0 **TX9D**：发送数据的第 9 位

该位可以是地址 / 数据位或奇偶校验位。

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

寄存器 16-2： RCSTA1：EUSART 接收状态和控制寄存器

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-0	R-0	R-x
SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D
bit 7							bit 0

bit 7 **SPEN**：串口使能位
1 = 使能串口 （配置 RX/DT 和 TX/CK 引脚作为串口引脚）
0 = 禁止串口 （保持在复位状态）

bit 6 **RX9**：9 位接收使能位
1 = 选择 9 位接收
0 = 选择 8 位接收

bit 5 **SREN**：单字节接收使能位
异步模式：
忽略。
同步主控模式：
1 = 使能单字节接收
0 = 禁止单字节接收
此位在接收完成后清零。
同步从动模式：
忽略。

bit 4 **CREN**：连续接收使能位
异步模式：
1 = 使能接收器
0 = 禁止接收器
同步模式：
1 = 使能连续接收，直到使能位 CREN 清零 （CREN 比 SREN 优先级高）
0 = 禁止连续接收

bit 3 **ADDEN**：地址检测使能位
9 位异步模式 （RX9 = 1）：
1 = 当 RSR<8> 置 1 时，使能地址检测、允许中断和装入接收缓冲器
0 = 禁止地址检测、接收所有字节并且第 9 位可用作奇偶校验位
9 位异步模式 （RX9 = 0）：
忽略。

bit2 **FERR**：帧错误位
1 = 帧错误 （可以通过读 RCREG 寄存器刷新该位并接收下一个有效字节）
0 = 无帧错误

bit1 **OERR**：溢出错误位
1 = 溢出错误 （可以通过清除 CREN 位清零）
0 = 无溢出错误

bit0 **RX9D**：接收数据的第 9 位
该位可以是地址 / 数据位或奇偶校验位，并且必须由用户固件计算得到。

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

寄存器 16-3： BAUDCON1：波特率控制寄存器 1

R/W-0	R-1	U-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0
ABDOVF	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN
bit 7							bit 0

bit 7 **ABDOVF**：自动波特采样进位状态位
1 = 在自动波特率检测模式下出现了 BRG 进位
（必须用软件清零）
0 = 没有发生 BRG 进位

bit 6 **RCIDL**：接收操作空闲状态位
1 = 接收操作处于空闲状态
0 = 接收操作处于活动状态

bit 5 未用位：读为 0

bit 4 **SCKP**：同步时钟极性选择位
异步模式：
在此模式下未使用。
同步模式：
1 = 空闲状态时钟 （CK）为高电平
0 = 空闲状态时钟 （CK）为低电平

bit 3 **BRG16**：16 位波特率寄存器使能位
1 = 16 位波特率发生器——SPBRGH1 和 SPBRG1
0 = 8 位波特率发生器——仅 SPBRG1 （兼容模式），忽略 SPBRGH1 的值

bit 2 未用位：读为 0

bit 1 **WUE**：唤醒使能位
异步模式：
1 = EUSART 将继续采样 RX 引脚——中断在下降沿产生，在下一个上升沿由硬件清零该位
0 = 未监测 RX 引脚或检测到了上升沿
同步模式：
在此模式下未使用

bit 0 **ABDEN**：自动波特率检测使能位
异步模式：
1 = 在下一个字符使能波特率检测。需要收到 “同步”字段 （55h），完成时在硬件中清零。
0 = 禁止波特率检测或检测已完成
同步模式：
在此模式下未使用。

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

16.1 EUSART 波特率发生器 （BRG）

BRG 是一个专用的 8 位或 16 位发生器，支持 EUSART 的异步和同步模式。默认情况下，BRG 工作在 8 位模式下，将 BRG 的第 16 位 （BAUDCON1<3>）置 1 可选择 16 位模式。

SPBRGH1:SPBRG1 寄存器对控制独立运行定时器的周期。在异步模式下， BRGH （TXSTA<2>）和 BRG16 （BAUDCON1<3>）也用于控制波特率。在同步模式下，BRGH 位会被忽略。表 16-1 所示为不同 EUSART 模式的波特率计算公式，但仅适用于主控模式 （由内部产生时钟信号）。

在给定目标波特率和 FOSC 的情况下，可以使用表 16-1 中的公式计算SPBRGH1:SPBRG1寄存器中的最近整数值，从而判断波特率误差。例 16-1 给出了计算示例。表 16-2 中给出了不同异步模式下典型的波特率和误差值。使用高波特率 （BRGH = 1）或 16 位 BRG 有利于减小波特率误差，或者在快速振荡频率条件下实现低波特率。

向 SSPBRGH1:SPBRG1 寄存器写入新值会引起 BRG 定时器复位 （或清零）。这可以确保 BRG 无需等待定时器溢出就可以输出新的波特率。

16.1.1 在功耗管理模式下的操作

器件时钟用于产生所需的波特率。当进入一种功耗管理模式时，新时钟源可能会工作在一个不同的频率下。这可能需要调整 SPBRG1 寄存器对中的值。

16.1.2 采样

检测电路对 RX/DT 引脚采样三次，以判定 RX 引脚上出现的是高电平还是低电平。

表 16-1： 波特率公式

配置位			BRG/EUSART 模式	波特率计算公式
SYNC	BRG16	BRGH		
0	0	0	8 位 / 异步	FOSC/[64 (n + 1)]
0	0	1	8 位 / 异步	FOSC/[16 (n + 1)]
0	1	0	16 位 / 异步	
0	1	1	16 位 / 异步	FOSC/[4 (n + 1)]
1	0	x	8 位 / 同步	
1	1	x	16 位 / 同步	

图注： x = 可忽略的值， n = SPBRGH1:SPBRG1 寄存器对的值

例 16-1： 计算波特率误差

器件工作在 FOSC = 16 MHz，目标波特率 = 9600 bps，异步模式， 8 位 BRG：
目标波特率 = FOSC/(64 ([SPBRGH1:SPBRG1] + 1))
求解 SPBRGH1:SPBRG1：
X = ((FOSC/ 目标波特率)/64) – 1
= ((16000000/9600)/64) – 1
= [25.042] = 25
波特率计算结果 = 16000000/(64 (25 + 1))
= 9615
误差 = (波特率计算结果 – 目标波特率) / 目标波特率
= (9615 – 9600)/9600 = 0.16%

表 16-2： 与波特率发生器相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
TXSTA1	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	61
RCSTA1	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	61
BAUDCON1	ABDOVF	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	62
SPBRGH1	EUSART1 波特率发生器寄存器的高字节								62
SPBRG1	EUSART1 波特率发生器寄存器的低字节								61

图注： — = 未用位，读为 0。 BRG 未使用阴影单元。

表 **16-3**： 异步模式下的波特率

目标波特率（Kbps）	SYNC = 0, BRGH = 0, BRG16 = 0											
	Fosc = 40.000 MHz			Fosc = 20.000 MHz			Fosc = 10.000 MHz			Fosc = 8.000 MHz		
	实际波特率（Kbps）	误差 %	SPBRG 值（10 进制）	实际波特率（Kbps）	误差 %	SPBRG 值（10 进制）	实际波特率（Kbps）	误差 %	SPBRG 值（10 进制）	实际波特率（Kbps）	误差 %	SPBRG 值（10 进制）
0.3	—	—	—	—	—	—	—	—	—	—	—	—
1.2	—	—	—	1.221	1.73	255	1.202	0.16	129	1.201	-0.16	103
2.4	2.441	1.73	255	2.404	0.16	129	2.404	0.16	64	2.403	-0.16	51
9.6	9.615	0.16	64	9.766	1.73	31	9.766	1.73	15	9.615	-0.16	12
19.2	19.531	1.73	31	19.531	1.73	15	19.531	1.73	7	—	—	—
57.6	56.818	-1.36	10	62.500	8.51	4	52.083	-9.58	2	—	—	—
115.2	125.000	8.51	4	104.167	-9.58	2	78.125	-32.18	1	—	—	—

目标波特率（Kbps）	SYNC = 0, BRGH = 0, BRG16 = 0								
	Fosc = 4.000 MHz			Fosc = 2.000 MHz			Fosc = 1.000 MHz		
	实际波特率（Kbps）	误差 %	SPBRG 值（10 进制）	实际波特率（Kbps）	误差 %	SPBRG 值（10 进制）	实际波特率（Kbps）	误差 %	SPBRG 值（10 进制）
0.3	0.300	0.16	207	0.300	-0.16	103	0.300	-0.16	51
1.2	1.202	0.16	51	1.201	-0.16	25	1.201	-0.16	12
2.4	2.404	0.16	25	2.403	-0.16	12	—	—	—
9.6	8.929	-6.99	6	—	—	—	—	—	—
19.2	20.833	8.51	2	—	—	—	—	—	—
57.6	62.500	8.51	0	—	—	—	—	—	—
115.2	62.500	-45.75	0	—	—	—	—	—	—

目标波特率（Kbps）	SYNC = 0, BRGH = 1, BRG16 = 0											
	Fosc = 40.000 MHz			Fosc = 20.000 MHz			Fosc = 10.000 MHz			Fosc = 8.000 MHz		
	实际波特率（Kbps）	误差 %	SPBRG 值（10 进制）	实际波特率（Kbps）	误差 %	SPBRG 值（10 进制）	实际波特率（Kbps）	误差 %	SPBRG 值（10 进制）	实际波特率（Kbps）	误差 %	SPBRG 值（10 进制）
0.3	—	—	—	—	—	—	—	—	—	—	—	—
1.2	—	—	—	—	—	—	—	—	—	—	—	—
2.4	—	—	—	—	—	—	2.441	1.73	255	2.403	-0.16	207
9.6	9.766	1.73	255	9.615	0.16	129	9.615	0.16	64	9.615	-0.16	51
19.2	19.231	0.16	129	19.231	0.16	64	19.531	1.73	31	19.230	-0.16	25
57.6	58.140	0.94	42	56.818	-1.36	21	56.818	-1.36	10	55.555	3.55	8
115.2	113.636	-1.36	21	113.636	-1.36	10	125.000	8.51	4	—	—	—

目标波特率（Kbps）	SYNC = 0, BRGH = 1, BRG16 = 0								
	Fosc = 4.000 MHz			Fosc = 2.000 MHz			Fosc = 1.000 MHz		
	实际波特率（Kbps）	误差 %	SPBRG 值（10 进制）	实际波特率（Kbps）	误差 %	SPBRG 值（10 进制）	实际波特率（Kbps）	误差 %	SPBRG 值（10 进制）
0.3	—	—	—	—	—	—	0.300	-0.16	207
1.2	1.202	0.16	207	1.201	-0.16	103	1.201	-0.16	51
2.4	2.404	0.16	103	2.403	-0.16	51	2.403	-0.16	25
9.6	9.615	0.16	25	9.615	-0.16	12	—	—	—
19.2	19.231	0.16	12	—	—	—	—	—	—
57.6	62.500	8.51	3	—	—	—	—	—	—
115.2	125.000	8.51	1	—	—	—	—	—	—

表 16-3： 异步模式下的波特率 （续）

目标波特率（Kbps）	SYNC = 0, BRGH = 0, BRG16 = 1											
	Fosc = 40.000 MHz			Fosc = 20.000 MHz			Fosc = 10.000 MHz			Fosc = 8.000 MHz		
	实际波特率（Kbps）	误差 %	SPBRG 值（10 进制）	实际波特率（Kbps）	误差 %	SPBRG 值（10 进制）	实际波特率（Kbps）	误差 %	SPBRG 值（10 进制）	实际波特率（Kbps）	误差 %	SPBRG 值（10 进制）
0.3	0.300	0.00	8332	0.300	0.02	4165	0.300	0.02	2082	0.300	-0.04	1665
1.2	1.200	0.02	2082	1.200	-0.03	1041	1.200	-0.03	520	1.201	-0.16	415
2.4	2.402	0.06	1040	2.399	-0.03	520	2.404	0.16	259	2.403	-0.16	207
9.6	9.615	0.16	259	9.615	0.16	129	9.615	0.16	64	9.615	-0.16	51
19.2	19.231	0.16	129	19.231	0.16	64	19.531	1.73	31	19.230	-0.16	25
57.6	58.140	0.94	42	56.818	-1.36	21	56.818	-1.36	10	55.555	3.55	8
115.2	113.636	-1.36	21	113.636	-1.36	10	125.000	8.51	4	—	—	—

目标波特率（Kbps）	SYNC = 0, BRGH = 0, BRG16 = 1								
	Fosc = 4.000 MHz			Fosc = 2.000 MHz			Fosc = 1.000 MHz		
	实际波特率（Kbps）	误差 %	SPBRG 值（10 进制）	实际波特率（Kbps）	误差 %	SPBRG 值（10 进制）	实际波特率（Kbps）	误差 %	SPBRG 值（10 进制）
0.3	0.300	0.04	832	0.300	-0.16	415	0.300	-0.16	207
1.2	1.202	0.16	207	1.201	-0.16	103	1.201	-0.16	51
2.4	2.404	0.16	103	2.403	-0.16	51	2.403	-0.16	25
9.6	9.615	0.16	25	9.615	-0.16	12	—	—	—
19.2	19.231	0.16	12	—	—	—	—	—	—
57.6	62.500	8.51	3	—	—	—	—	—	—
115.2	125.000	8.51	1	—	—	—	—	—	—

目标波特率（Kbps）	SYNC = 0, BRGH = 1, BRG16 = 1 或 SYNC = 1, BRG16 = 1											
	Fosc = 40.000 MHz			Fosc = 20.000 MHz			Fosc = 10.000 MHz			Fosc = 8.000 MHz		
	实际波特率（Kbps）	误差 %	SPBRG 值（10 进制）	实际波特率（Kbps）	误差 %	SPBRG 值（10 进制）	实际波特率（Kbps）	误差 %	SPBRG 值（10 进制）	实际波特率（Kbps）	误差 %	SPBRG 值（10 进制）
0.3	0.300	0.00	33332	0.300	0.00	16665	0.300	0.00	8332	0.300	-0.01	6665
1.2	1.200	0.00	8332	1.200	0.02	4165	1.200	0.02	2082	1.200	-0.04	1665
2.4	2.400	0.02	4165	2.400	0.02	2082	2.402	0.06	1040	2.400	-0.04	832
9.6	9.606	0.06	1040	9.596	-0.03	520	9.615	0.16	259	9.615	-0.16	207
19.2	19.193	-0.03	520	19.231	0.16	259	19.231	0.16	129	19.230	-0.16	103
57.6	57.803	0.35	172	57.471	-0.22	86	58.140	0.94	42	57.142	0.79	34
115.2	114.943	-0.22	86	116.279	0.94	42	113.636	-1.36	21	117.647	-2.12	16

目标波特率（Kbps）	SYNC = 0, BRGH = 1, BRG16 = 1 或 SYNC = 1, BRG16 = 1								
	Fosc = 4.000 MHz			Fosc = 2.000 MHz			Fosc = 1.000 MHz		
	实际波特率（Kbps）	误差 %	SPBRG 值（10 进制）	实际波特率（Kbps）	误差 %	SPBRG 值（10 进制）	实际波特率（Kbps）	误差 %	SPBRG 值（10 进制）
0.3	0.300	0.01	3332	0.300	-0.04	1665	0.300	-0.04	832
1.2	1.200	0.04	832	1.201	-0.16	415	1.201	-0.16	207
2.4	2.404	0.16	415	2.403	-0.16	207	2.403	-0.16	103
9.6	9.615	0.16	103	9.615	-0.16	51	9.615	-0.16	25
19.2	19.231	0.16	51	19.230	-0.16	25	19.230	-0.16	12
57.6	58.824	2.12	16	55.555	3.55	8	—	—	—
115.2	111.111	-3.55	8	—	—	—	—	—	—

16.1.3 自动波特率检测

增强型 USART 模块支持波特率自动检测和校准。此功能仅在异步模式下当 WUE 位清零时有效。

只要接收到起始位并且 ABDEN 位已置 1，就启动自动波特率检测 （图 16-1） 。波特率计算采用自平均的方式。

在自动波特率检测（Auto-Baud Rate Detect，ABD）模式下， BRG 的时钟是反向的。不是由 BRG 为 RX 输入信号提供时钟源，而是由 RX 信号为 BRG 定时。在 ABD 模式下，内部波特率发生器被用作计数器来计算输入的串行字节流的位间隔时间。

一旦 ABDEN 位置 1，状态机就会将 BRG 清零并寻找起始位。为了正确计算比特率，自动波特率检测必须接收到一个值为 55h （ASCII 字符 U，也是 LIN 总线的同步字符）的字节。为了尽量减少输入信号不对称造成的影响，在接收低位和高位的时间内都要进行测量。在起始位后，SPBRG 使用预先选择的时钟源在 RX1 的第一个上升沿开始计数。在 RX 引脚传输了 8 个位，或在检测到第 5 个上升沿后，会将相应 BRG 周期内的累加值保存在 SPBRGH1:SPBRG1 寄存器对中。当第 5 个时钟周期出现时 （应与停止位对应）， ABDEN 位会自动清零。

如果发生了 BRG 计满返回 （从 FFFFh 到 0000h 的溢出），会在 ABDOVF 状态位 （BAUDCON1<7>）有所反映。该位可在 BRG 进位时由硬件置 1，也可以由用户通过软件置1或清零。在发生进位事件后，继续保持ABD 模式， ABDEN 位保持置 1 （图 16-2）。

在校准波特率周期时，BRG寄存器时钟频率为预配置时钟频率的 1/8。请注意 BRG 时钟将由 BRG16 和 BRGH 位配置。无论 BRG16 的设置如何， SPBRG1 和 SPBRGH1 将被用作一个 16 位计数器。用户通过检查 SPBRGH1 寄存器的值是否为 00h，可以验证 8 位模式下是否发生了进位。表 16-4 所示为 BRG 计数器的时钟速率。

当产生ABD时序时，EUSART状态机保持在空闲状态。一旦在 RX1 上检测到第 5 个上升沿，中断标志位 RC1IF 就会置 1。需要读取 RCREG1 中的值，来清除中断标志位 RC1IF。应丢弃 RCREG1 的值。

注 1：如果 WUE 位与 ABDEN 位同时置 1，自动波特率检测会在间隔字符*之后*的字节开始。

2：判断进入的字符波特率是否处于所选的 BRG 时钟源范围内是由用户决定的。由于位错误率的原因，某些振荡频率和 EUSART 波特率的组合是无法实现的。在使用自动波特率检测功能时，必须综合考虑系统总的时序和通信波特率。

表 16-4： BRG 计数器时钟速率

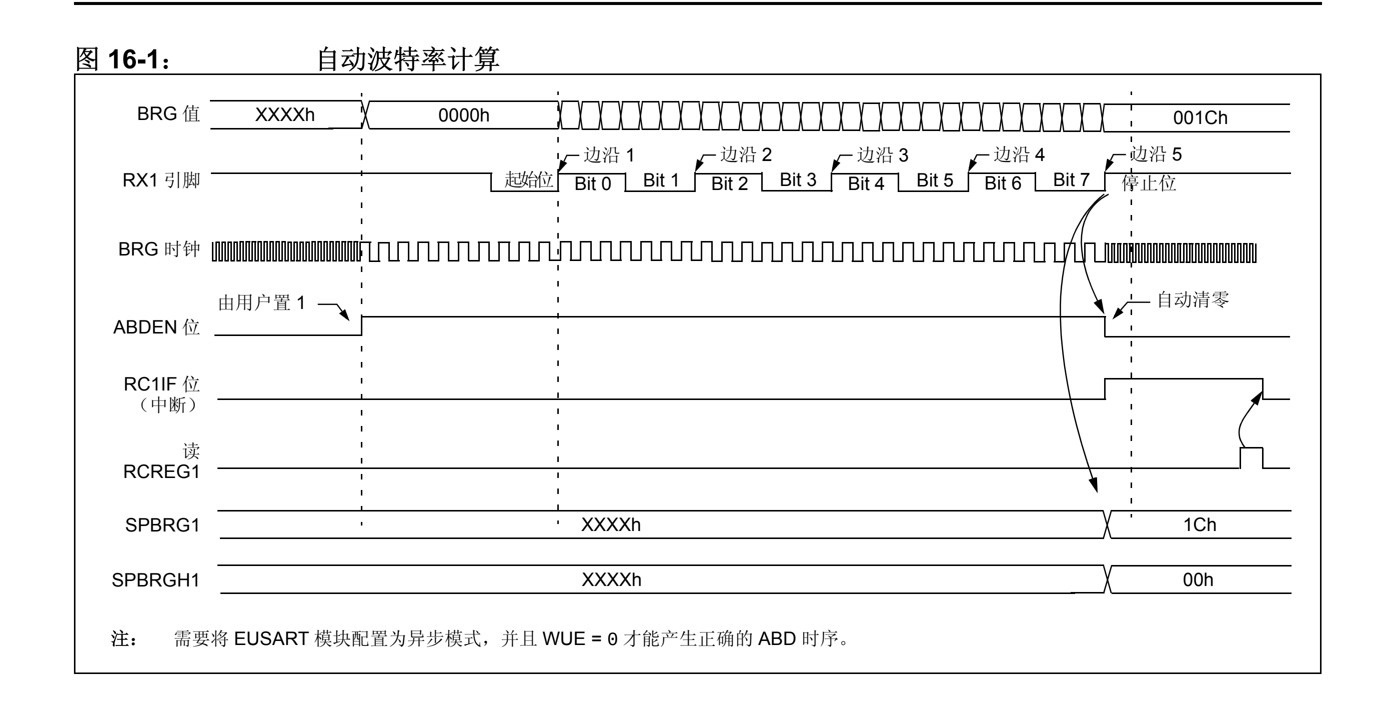
BRG16	BRGH	BRG 计数器时钟
0	0	Fosc/512
0	1	Fosc/128
1	0	Fosc/128
1	1	Fosc/32

注： 在产生 ABD 时序时，不管 BRG16 如何设置，SPBRG1 和 SPBRGH1 被用作一个 16 位计数器。

16.1.3.1 ABD 和 EUSART 发送

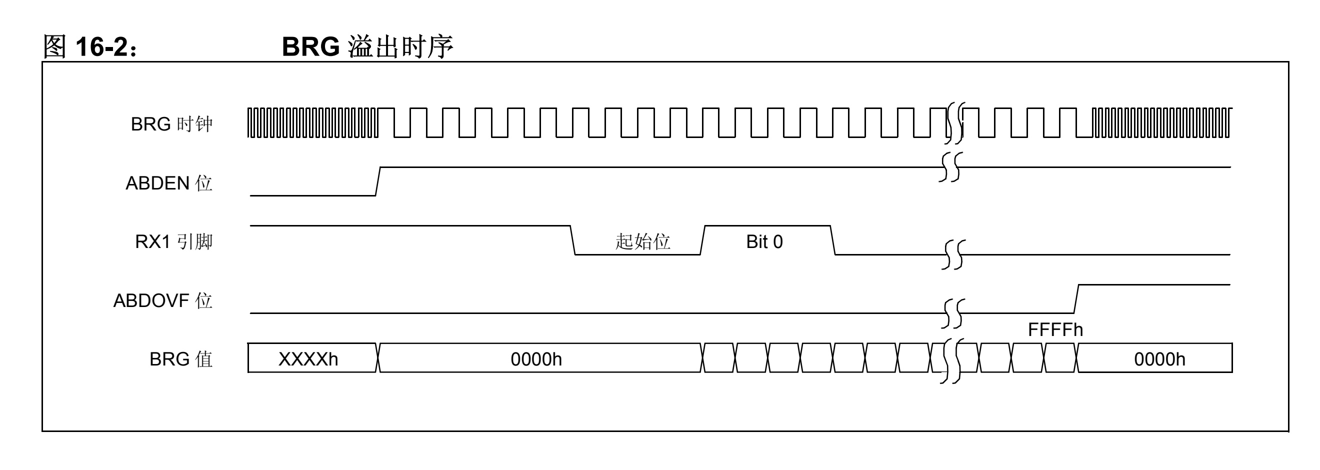
由于 ABD 采样期间 BRG 时钟是反向的，因此在 ABD 期间不能使用 EUSART 发送器。这意味着无论 ABDEN 位在什么时候置 1，都不能写入 TXREG1。用户还应确保在发送期间 ABDEN 不能为置 1 状态，否则可能会导致无法预料的 EUSART 操作。

图 16-1： 自动波特率计算



注： 需要将 EUSART 模块配置为异步模式，并且 WUE = 0 才能产生正确的 ABD 时序。

图 16-2： **BRG** 溢出时序



16.2 EUSART 异步模式

通过将 SYNC （TXSTA1<4>）位清零可选择异步工作模式。在此模式下， EUSART 使用标准的不归零（Non-Return-to-Zero，NRZ）格式（1 个起始位、8 个或 9 个数据位和 1 个停止位）。最常用的数据格式为 8 位。片上专用 8 位 /16 位波特率发生器可借助于振荡器产生标准波特率频率。

EUSART 首先发送和接收的是最低有效位。EUSART 的发送器和接收器在功能上是独立的，但采用相同的数据格式和波特率。波特率发生器可以根据 BRGH 位和 BRG16 位 （TXSTA1<2> 和 BAUDCON1<3>）的设置值产生两种不同的波特率时钟，频率分别为移位速率的 16 倍或 64 倍。硬件不支持奇偶校验，但可以用软件实现，校验位保存在第 9 个数据位中。

当工作在异步模式下时，EUSART 模块包括以下重要组成部分：

- 波特率发生器
- 采样电路
- 异步发送器
- 异步接收器
- 同步间隔字符自动唤醒
- 12 位间隔字符发送
- 自动波特率检测

16.2.1 EUSART 异步发送器

图 16-3 显示了 EUSART 发送器的框图。发送器的核心是发送（串行）移位寄存器（Transmit Shift Register，TSR）。移位寄存器从读 / 写发送缓冲寄存器 TXREG1 中获取数据。TXREG1 寄存器中的数据由软件写入。在前一次装入的停止位发送前，不会向 TSR 寄存器装入数据。一旦停止位发送完毕，TXREG1 寄存器中的新数据（如果有的话）就会被装入 TSR。

一旦 TXREG1 寄存器向 TSR 寄存器传输了数据 （在 1 个 TCY 内发生），TXREG1 寄存器就为空，同时标志位 TX1IF（PIR1<4>）置 1。可以通过将中断使能位 TX1IE（PIE1<4>）置1或清零来使能/禁止该中断。不管TX1IE 的状态如何，只要中断发生，TX1IF 就会置 1 并且不能用软件清零。TX1IF 不会在 TXREG1 装入新数据时立即被清零，而是在装入指令后的第二个指令周期被清零。因此在 TXREG1 装入新数据后立即查询 TX1IF，会得到无效结果。

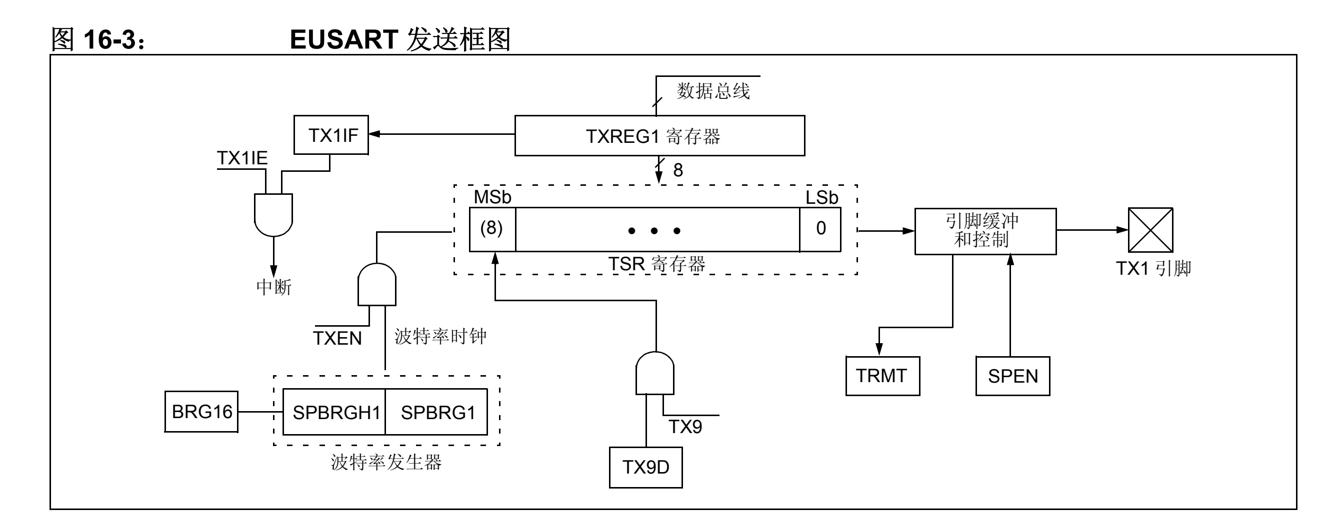
TX1IF 指示的是 TXREG1 寄存器的状态，而另一个位 TRMT （TXSTA1<1>）则指示 TSR 寄存器的状态。TRMT 是只读位，它在 TSR 寄存器为空时被置 1。TRMT 位与任何中断均无关联，因此要确定 TSR 寄存器是否为空，用户只能对此位进行轮询。

> **注 1：** TSR 寄存器并未映射到数据存储器中，因此用户不能直接访问它。
>
> **2：** 当使能位 TXEN 置 1 时，标志位 TX1IF 也会置 1。

设置异步发送操作的步骤如下：

1. 对 SPBRGH1:SPBRG1 寄存器进行初始化，设置合适的波特率。按需要将 BRGH 和 BRG16 位置 1 或清零，以获得目标波特率。
2. 将 SYNC 位清零、SPEN 位置 1，使能异步串口。
3. 若需要中断，将中断使能位 TX1IE 置 1
4. 若需要发送 9 位数据，将发送位 TX9 置 1。发送的第 9 位可以是地址位也可以是数据位。
5. 通过将 TXEN 位置 1 使能发送，此操作同时也会将 TX1IF 位置 1。
6. 如果选择发送 9 位数据，应该将第 9 位数据装入 TX9D 位。
7. 将数据装入 TXREG1 寄存器 （开始发送）。
8. 若想使用中断，请确保将 INTCON 寄存器中的 GIE 和 PEIE 位 （INTCON<7:6>）被置 1。

图 16-3： EUSART 发送框图



 初稿

图 16-4： 异步发送时序

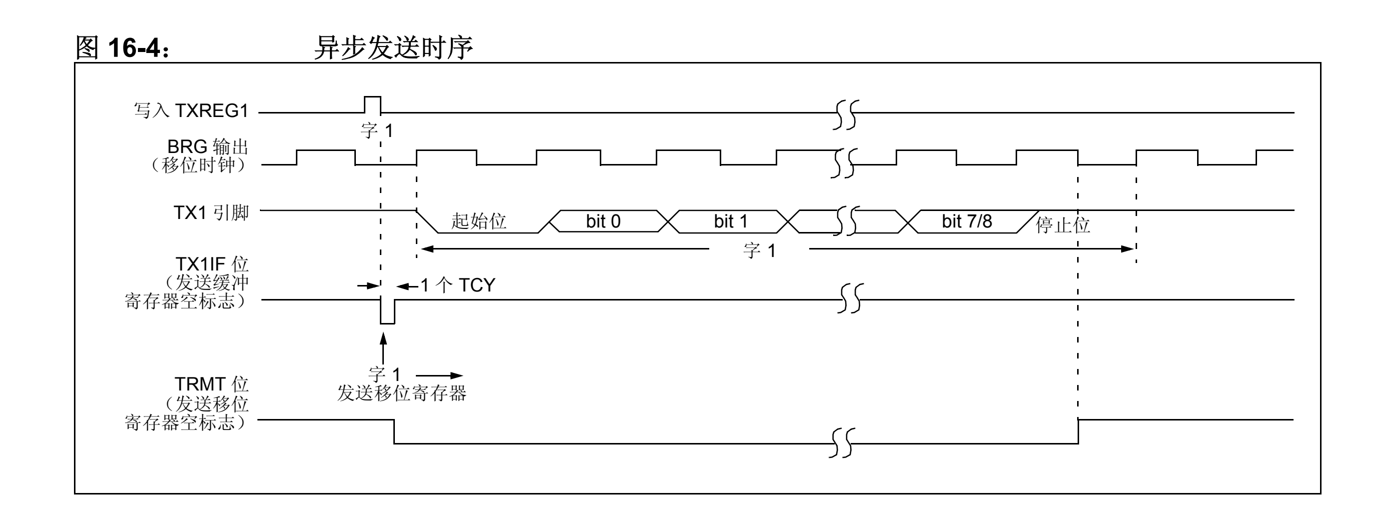


图 16-5： 异步发送时序 （背对背）

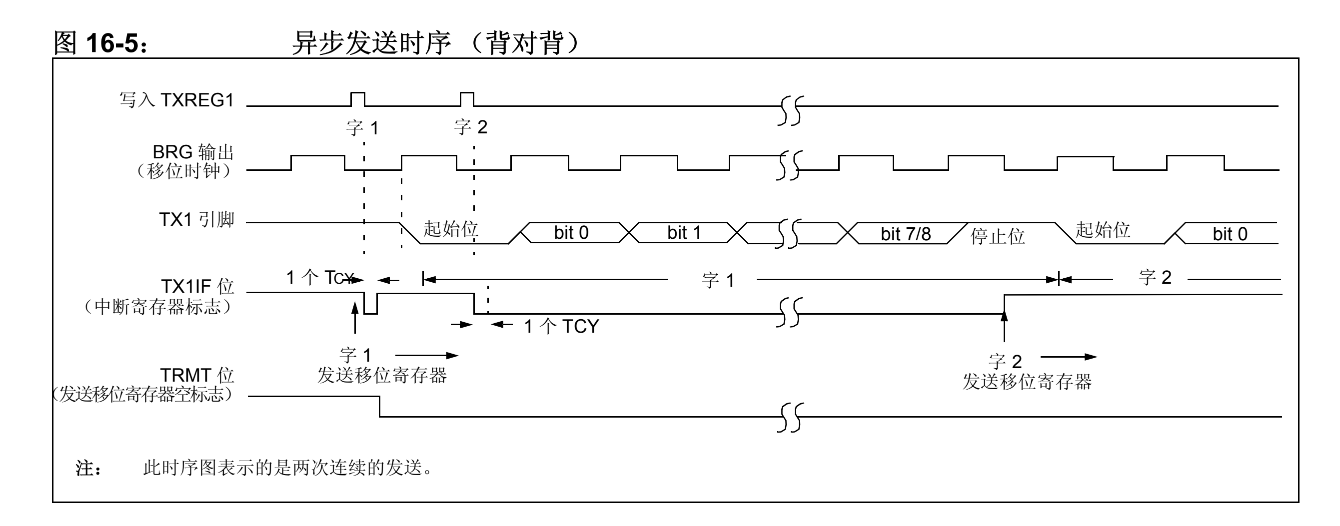


表 16-5： 与异步发送相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
PIR1	—	ADIF	RC1IF	TX1IF	SSPIF	CCP1IF	TMR2IF	TMR1IF	61
PIE1	—	ADIE	RC1IE	TX1IE	SSPIE	CCP1IE	TMR2IE	TMR1IE	61
IPR1	—	ADIP	RC1IP	TX1IP	SSPIP	CCP1IP	TMR2IP	TMR1IP	61
RCSTA1	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	61
TXREG1	EUSART1 发送寄存器								61
TXSTA1	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	61
BAUDCON1	ABDOVF	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	62
SPBRGH1	EUSART1 波特率发生器寄存器的高字节								62
SPBRG1	EUSART1 波特率发生器寄存器的低字节								61

图注： — = 未用位，读为 0。异步发送不使用阴影单元。

16.2.2 EUSART 异步接收器

图 16-6 显示了接收器的原理框图。在 RX1 引脚上接收数据，并驱动数据恢复电路。数据恢复电路实际上是一个工作频率为 16 倍波特率的高速移位器，而主接收串行移位器的工作频率等于比特率或 FOSC。此模式通常用于 RS-232 系统。

设置异步接收操作的步骤如下：

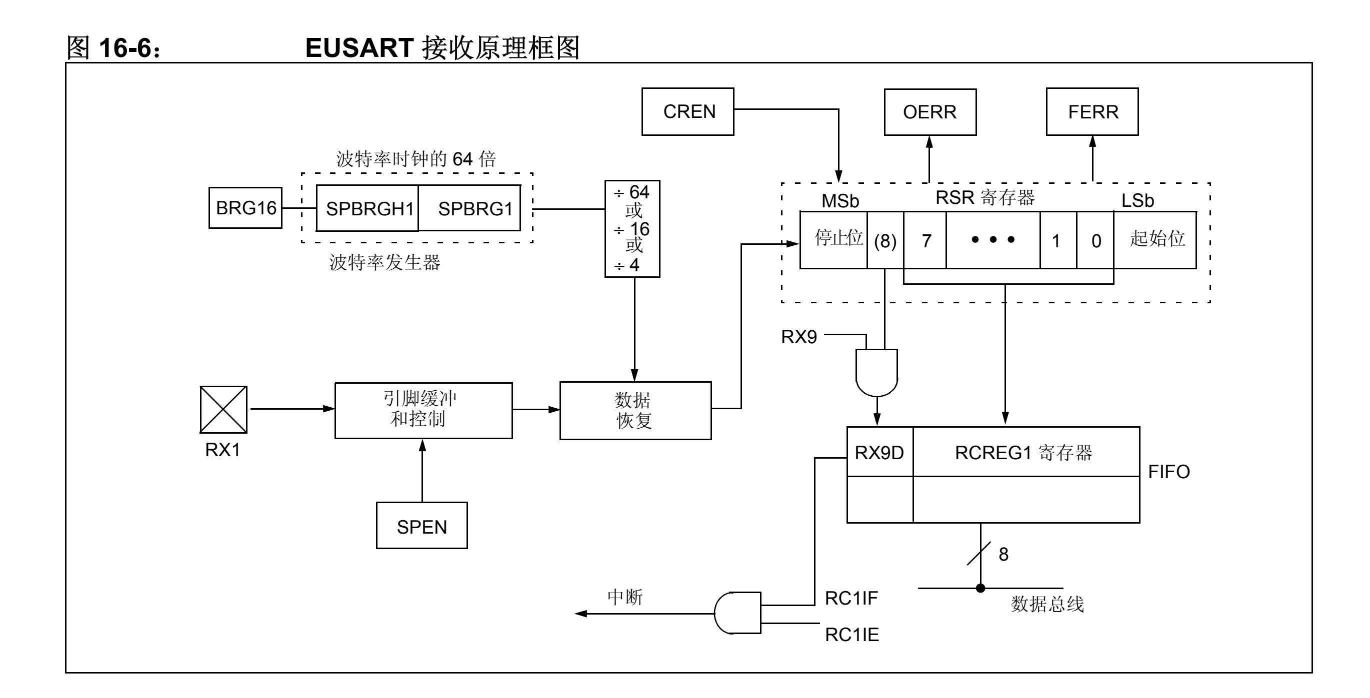
1. 对 SPBRGH1:SPBRG1 寄存器进行初始化，设置合适的波特率。按需要将 BRGH 和 BRG16 位置 1 或清零，以获得目标波特率。
2. 通过将 SYNC 位清零、SPEN 位置 1，使能异步串口。
3. 若需要中断，将中断使能位 RC1IE 置 1。
4. 若需要接收 9 位数据，将 RX9 位置 1。
5. 通过将 CREN 位置 1，使能接收。
6. 当接收完成时标志位 RC1IF 将被置 1，此时如果中断使能位 RC1IE 已置 1，还将产生一个中断。
7. 读 RCSTA1 寄存器获取第 9 位数据 （如果已使能），并判断在接收过程中是否发生了错误。
8. 通过读 RCREG1 寄存器来读取接收到的 8 位数据。
9. 如果发生错误，通过将使能位 CREN 清零来清除错误。
10. 若想使用中断，请确保将 INTCON 寄存器中的 GIE 和 PEIE 位 （INTCON<7:6>）置 1。

16.2.3 设置带有地址检测功能的 9 位模式

此模式通常用在 RS-485 系统中。按如下步骤设置带有地址检测功能的异步接收操作：

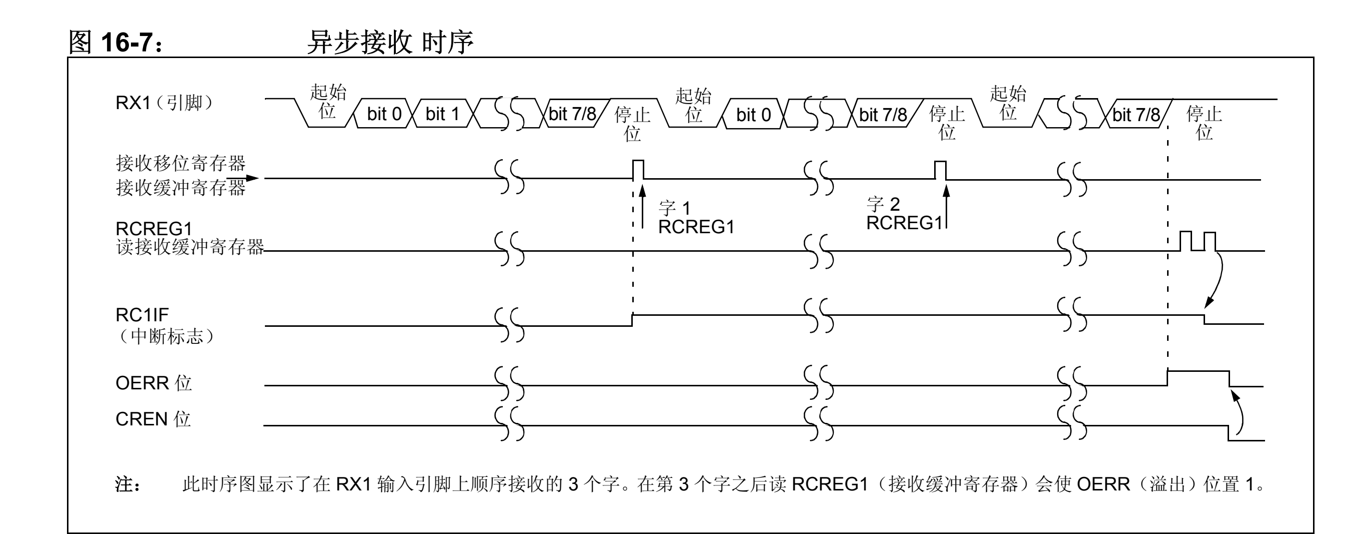
1. 初始化 SPBRGH1:SPBRG1 寄存器，设置合适的波特率。按需要将 BRGH 和 BRG16 位置 1 或清零，以获得目标波特率。
2. 通过将 SYNC 位清零、SPEN 位置 1，使能异步串口。
3. 若需要中断，将 RCEN 位置 1 并使用 RC1IP 位设置优先级。
4. 将 RX9 位置 1，使能 9 位接收。
5. 将 ADDEN 位置 1，使能地址检测。
6. 将 CREN 位置 1，使能接收。
7. 当接收完成时 RC1IF 位将被置 1。此时如果 RC1IE 和 GIE 位已置 1，还将响应中断。
8. 读 RCSTA1 寄存器判断在接收时是否发生了错误，同时读取第 9 位数据 （如果适用）。
9. 读 RCREG1 判断是否正在对器件进行寻址。
10. 如果发生错误，将 CREN 位清零。
11. 如果已经找到器件，将 ADDEN 位清零，允许接收到的数据进入接收缓冲器，并中断 CPU。

图 16-6： EUSART 接收原理框图



 初稿

图 16-7：　异步接收 时序



注：　此时序图显示了在 RX1 输入引脚上顺序接收的 3 个字。在第 3 个字之后读 RCREG1（接收缓冲寄存器）会使 OERR（溢出）位置 1。

表 16-6：　与异步接收相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
PIR1	—	ADIF	RC1IF	TX1IF	SSPIF	CCP1IF	TMR2IF	TMR1IF	61
PIE1	—	ADIE	RC1IE	TX1IE	SSPIE	CCP1IE	TMR2IE	TMR1IE	61
IPR1	—	ADIP	RC1IP	TX1IP	SSPIP	CCP1IP	TMR2IP	TMR1IP	61
RCSTA1	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	61
RCREG1	EUSART1 接收寄存器								61
TXSTA1	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	61
BAUDCON1	ABDOVF	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	62
SPBRGH1	EUSART1 波特率发生器寄存器的高字节								62
SPBRG1	EUSART1 波特率发生器寄存器的低字节								61

图注：　— = 未用位，读为 0。异步接收不使用阴影单元。

16.2.4 同步间隔字符自动唤醒

在休眠模式下，EUSART 的所有时钟都会暂停。因此，波特率发生器处于非激活状态，且无法进行正确的数据接收。自动唤醒功能允许当 RX1/DT1 线上有事件发生时唤醒控制器，该功能需要 EUSART 工作在异步模式下。

通过将 WUE 位 （BAUDCON<1>）置 1，使能自动唤醒功能。该功能启用后，将禁止 RX1/DT1 上的典型接收操作，且 EUSART 保持在空闲状态并监视唤醒事件（不管 CPU 运行模式如何）。唤醒事件是指 RX1/DT1 线上发生高电平到低电平的转换。（这与 “同步中断” 字符或 LIN 协议唤醒信号字符的启动条件一致。）

唤醒事件后，模块产生一个 RC1IF 中断。在正常工作模式下，中断会与 Q 时钟同步产生 （图 16-8）；如果器件处于休眠模式，则两者是不同步的（图 16-9）。通过读 RCREG1 寄存器可清除中断条件。

唤醒事件后，当 RX1 线上出现由低向高的电平转换时，WUE 位自动清零。此时， EUSART 模块将从空闲状态返回正常工作模式，由此用户可知 “同步中断” 事件已经结束。

16.2.4.1 使用自动唤醒功能的注意事项

因为自动唤醒功能是通过检测 RX1/DT1 上的上升沿跳变实现的，所以在停止位前该引脚上任何的状态改变都可能会产生错误的结束信号并导致数据或帧错误。因此，为了正确保正常的传输，必须首先发送全 0 字符。对于标准的 RS-232 器件，该字符是 00h （8 位），而对于 LIN 总线器件则是 000h （12 位）。

另外还必须考虑振荡器起振时间，尤其在采用起振间隔较长的振荡器 （即， XT 或 HS 模式）应用中更要注意这一点。“同步中断”（或唤醒信号）字符必须足够长，并且跟有足够长的时间间隔，以便使选定振荡器有充足的时间起振来保证 EUSART 正确初始化。

16.2.4.2 使用 WUE 位的注意事项

WUE 和 RC1IF 事件的时序用来判断接收数据的有效性时，有可能会引起一些混淆。如前所述，将 WUE 位置 1 会使 EUSART 进入空闲状态。唤醒事件会产生一个接收中断并将 RC1IF 位置 1。此后当 RX1/DT1 出现上升沿时，WUE 位被清零。然后通过读 RCREG1 寄存器清除中断条件。一般情况下，RCREG1 中的数据是无效数据，应该丢弃。

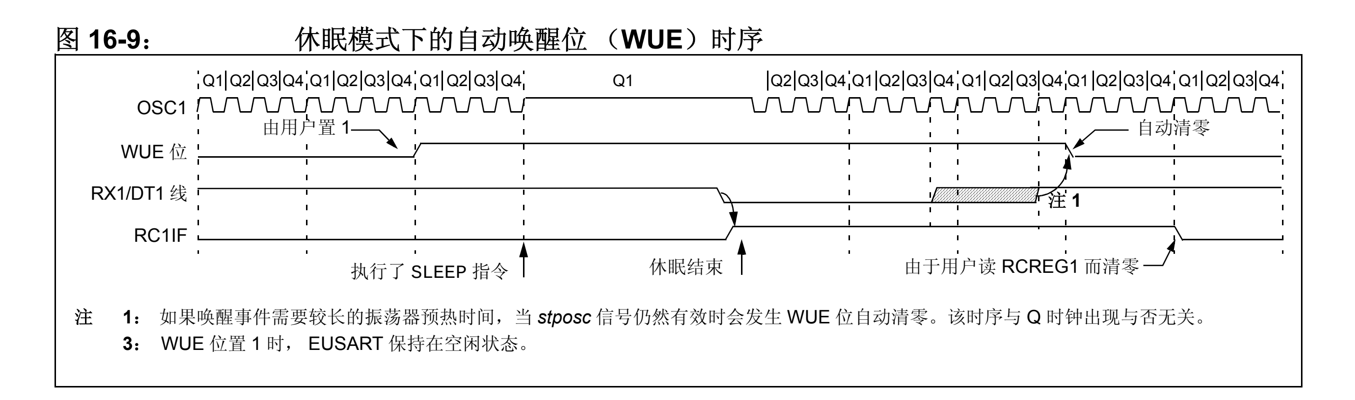
WUE 位清零 （或仍然置 1）且 RC1IF 标志位置 1 并不能表明 RCREG1 中数据接收是完整的。用户应该考虑在固件中同时验证是否完整地接收了数据。

要确保没有丢失有效数据，应检查RCIDL位来验证是否还在接收数据。如果不在接收数据，则可将 WUE 位置 1 ，使器件立即进入休眠模式。

图 16-8： 正常工作模式下的自动唤醒位 （**WUE**）时序

注： WUE 位置 1 时， EUSART 保持在空闲状态。

图 16-9： 休眠模式下的自动唤醒位 （**WUE**）时序



注 1： 如果唤醒事件需要较长的振荡器预热时间，当 *stposc* 信号仍然有效时会发生 WUE 位自动清零。该时序与 Q 时钟出现与否无关。
3： WUE 位置 1 时， EUSART 保持在空闲状态。

16.2.5 间隔字符序列

增强型 USART 模块能够发送符合 LIN 总线标准的特殊间隔字符。发送的间隔字符包括 1 个起始位，后面跟有 12 个 0 位和一个停止位。当发送移位寄存器装有数据时，只要SENDB和TXEN位（TXSTA<3>和TXSTA<5>）置 1，就会发送帧间隔字符。请注意写入 TXREG1 的数据值会被忽略，并会发送全 0。

在发送了相应的停止位后，硬件会自动将 SENDB 位清零。这样用户可以在间隔字符（在 LIN 规范中通常是同步字符）后将下一个要发送的字节预先装入发送 FIFO。

请注意写入 TXREG1 中作为间隔字符的数据值会被忽略。写入仅仅是为了启动正确的时序。

正如其在正常发送操作中一样，TRMT 位表明发送正在进行还是处于空闲状态。关于发送中断字符的时序，请参见图 16-10。

16.2.5.1 中断和同步发送序列

下述序列会发送一个报文帧头，包括一个间隔字符和其后的自动波特率同步字节。此序列适用于典型的 LIN 总线主控器件。

1. 将 EUSART 配置为所需的模式。
2. 将 TXEN 和 SENDB 位置 1，以设置间隔字符。
3. 将无效字符装入 TXREG1，启动发送（该值会被忽略）。
4. 将55h写入TXREG1，以便把同步字符装入FIFO 缓冲器。
5. 间隔字符发送后，硬件会将 SENDB 位复位。此时，同步字符会以预先配置的模式发送。

当 TXREG1 为空时（由 TX1IF 指出），下一个数据字节会写入 TXREG1。

16.2.6 接收间隔字符

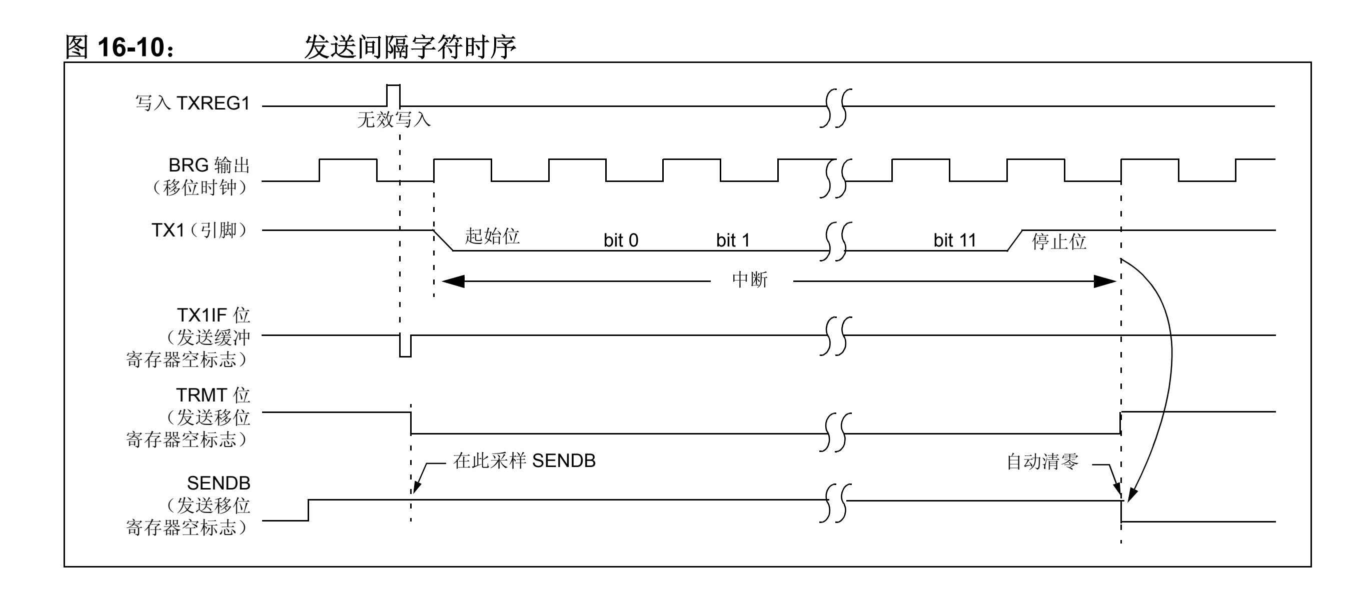
增强型 USART 模块接收间隔字符有两种方法。

第一种方法是强制将波特率配置为典型速率的 9/13。这可以使停止位在正确的采样点（对于间隔字符为起始位之后的 13 位，对于典型数据则是 8 个数据位）产生。

第二种方法是使用**第 16.2.4 节 "同步间隔字符自动唤醒"**中描述的自动唤醒功能。通过使能此功能，EUSART 将采样 RX1/DT1 上电平的下两次跳变，产生一个 RC1IF 中断，接收下一个数据字节，并在随后产生另一个中断。

请注意在间隔字符后，用户通常希望使能自动波特率检测功能。无论使用哪种方法，用户都可以在检测到 TX1IF 中断时马上将 ABD 位置 1。

图 16-10： 发送间隔字符时序



16.3 EUSART 同步主控模式

将 CSRC 位 （TXSTA<7>）置 1 可以进入同步主控模式。在此模式中，数据以半双工方式发送（即发送和接收不同时进行） 。发送数据时，禁止接收，反之亦然。将 SYNC 位 （TXSTA<4>）置 1 可进入同步模式。此外，应将使能位 SPEN （RCSTA1<7>）置 1，分别把 TX1 和 RX1 引脚配置为 CK1（时钟）和 DT1（数据）线。

主控模式意味着处理器在 CK1 时钟线上发送主控时钟信号。时钟极性是通过 SCKP 位 （BAUDCON<4>）选择的。将 SCKP 置 1 是将空闲状态时的 CK1 设为高电平，将该位清零则将空闲状态时的 CK1 设为低电平。此选项支持将本模块与 Microwire 器件配合使用。

16.3.1 EUSART 同步主控发送

图 16-3 显示了 EUSART 发送器的框图。发送器的核心是发送（串行）移位寄存器 （TSR）。该移位寄存器从读 / 写发送缓冲寄存器 TXREG1 中获取数据，而 TXREG1 寄存器中的数据由软件装入。在前一次装入数据的最后一位发送完成后，才向 TSR 寄存器装入新数据。一旦最后一位发送完成，就会将 TXREG1 寄存器中的新数据 （如果有的话）装入 TSR。

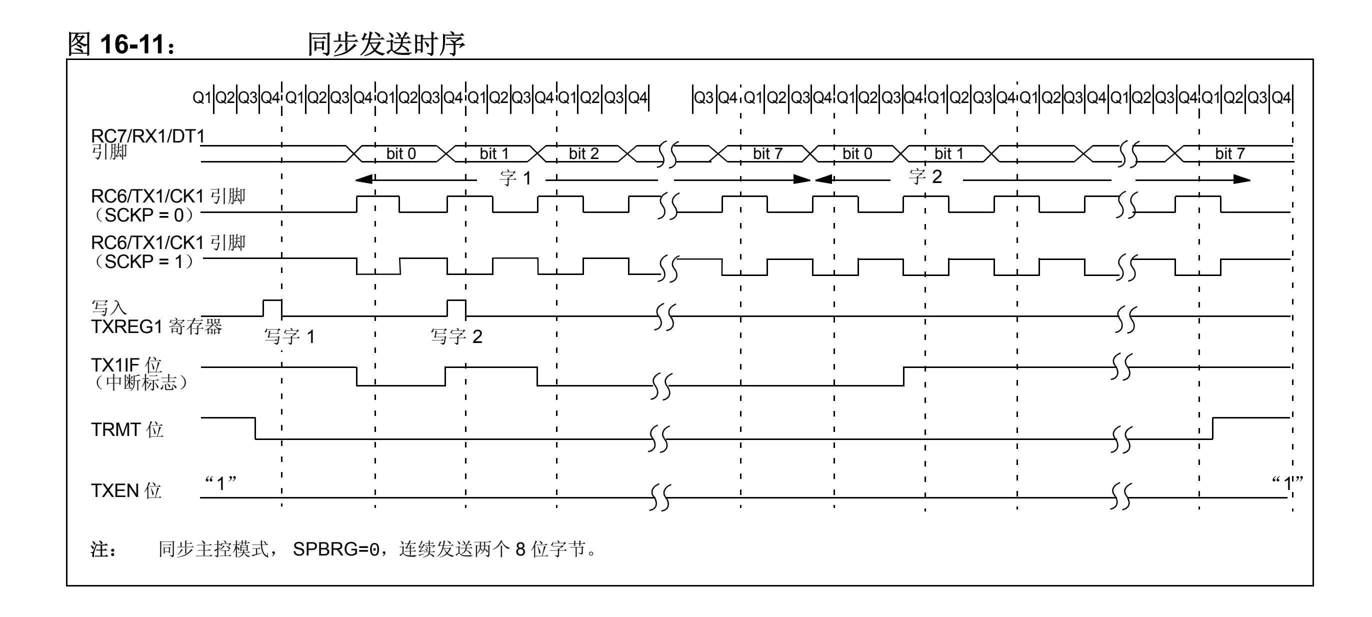
一旦 TXREG1 寄存器向 TSR 寄存器传输了数据 （在 1 个 TCY 内发生），TXREG1 寄存器就为空，同时标志位 TX1IF （PIR1<4>）被置 1。可以通过将中断使能位 TX1IE （PIE1<4>）置 1 或清零来使能或禁止该中断。TX1IF 的设置不受 TX1IE 状态的影响，且不能用软件清零。只有在新数据写入 TXREG1 寄存器时，TX1IF 才会复位。

TX1IF 表示的是 TXREG1 寄存器的状态，而另一个标志位 TRMT （TXSTA<1>）则表示 TSR 寄存器的状态。TRMT 位是一个只读位，当 TSR 为空时， TRMT 被置 1。 TRMT 位与任何中断均无关联，因此要判断 TSR 寄存器是否为空，用户只能对该位进行轮询。TSR 并未映射到数据存储器中，所以用户不能直接访问它。

设置同步主控发送操作的步骤如下：

1. 对 SPBRGH1:SPBRG1 寄存器进行初始化，设置合适的波特率。按需要将 BRG16 位置 1 或清零，以获得目标波特率。
2. 将 SYNC、SPEN 和 CSRC 位置 1，使能同步主控串口。
3. 若需要中断，将中断使能位 TX1IE 置 1。
4. 若需要发送 9 位数据，将 TX9 置 1。
5. 将 TXEN 位置 1，使能发送。
6. 如果选择发送 9 位数据，将第 9 位数据装入 TX9D 位。
7. 将数据装入 TXREG1 寄存器，启动发送。
8. 若想使用中断，请确保将 INTCON 寄存器中的 GIE 和 PEIE 位 （INTCON<7:6>）置 1。

图 16-11： 同步发送时序



 初稿

图 16-12： 同步发送时序 （由 **TXEN** 位控制）

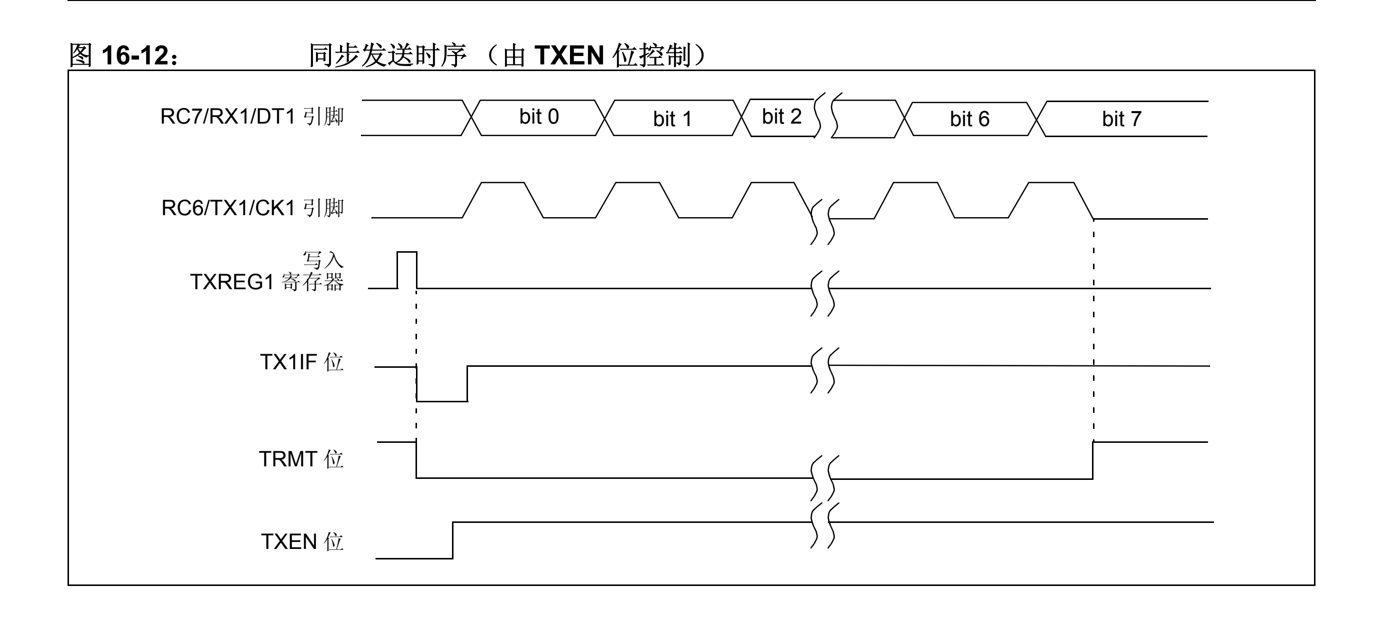


表 16-7： 与同步主控发送相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
PIR1	—	ADIF	RC1IF	TX1IF	SSPIF	CCP1IF	TMR2IF	TMR1IF	61
PIE1	—	ADIE	RC1IE	TX1IE	SSPIE	CCP1IE	TMR2IE	TMR1IE	61
IPR1	—	ADIP	RC1IP	TX1IP	SSPIP	CCP1IP	TMR2IP	TMR1IP	61
RCSTA1	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	61
TXREG1	EUSART1 发送寄存器								61
TXSTA1	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	61
BAUDCON1	ABDOVF	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	62
SPBRGH1	EUSART1 波特率发生器寄存器的高字节								62
SPBRG1	EUSART1 波特率发生器寄存器的低字节								61

图注： — = 未用位，读为 0。同步主控发送不使用阴影单元。

16.3.2 EAUSART 同步主控接收

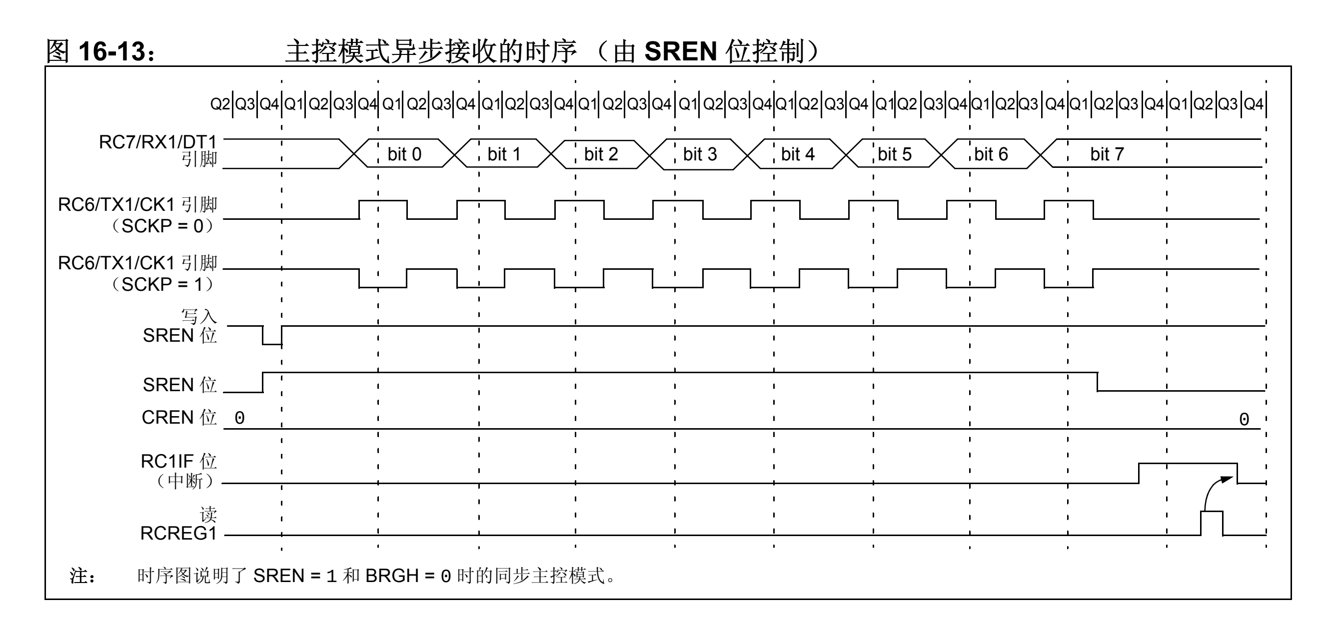
一旦选择了同步模式，只要将单字节接收使能位 SREN（RCSTA1<5>）或连续接收使能位 CREN（RCSTA1<4>）置 1，即可使能接收。在时钟的下降沿采样 RX1 引脚上的数据。

如果将使能位 SREN 置 1，则只接收单个字。如果将使能位 CREN 置 1，则会连续接收数据，直到将 CREN 位清零。如果两个位均被置 1，则 CREN 具有优先权。

设置同步主控接收操作的步骤如下：

1. 对 SPBRGH1:SPBRG1 寄存器进行初始化，设置合适的波特率。按需要将 BRG16 位置 1 或清零，以获得目标波特率。
2. 将 SYNC、SPEN 和 CSRC 位置 1，使能同步主控串口。
3. 确保将 CREN 和 SREN 位清零。
4. 若需要中断，将中断使能位 RC1IE 置 1。
5. 若需要接收 9 位数据，将 RX9 位置 1。
6. 若需要单字节接收，将 SREN 位置 1 ；若需要连续接收，将 CREN 位置 1。
7. 当接收完成时中断标志位 RC1IF 将置 1，此时如果中断使能位 RC1IE 已置 1，则还将产生一个中断。
8. 读 RCSTA1 寄存器以获取第 9 位数据（如果使能的话）并判断在接收过程中是否发生了错误。
9. 通过读 RCREG1 寄存器来读取接收到的 8 位数据。
10. 如果发生错误，将 CREN 位清零以清除错误。
11. 若想使用中断，请确保将 INTCON 寄存器中的 GIE 和 PEIE 位（INTCON<7:6>）置 1。

图 16-13： 主控模式异步接收的时序（由 **SREN** 位控制）



注： 时序图说明了 SREN = 1 和 BRGH = 0 时的同步主控模式。

表 16-8： 与同步主控接收相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
PIR1	—	ADIF	RC1IF	TX1IF	SSPIF	CCP1IF	TMR2IF	TMR1IF	61
PIE1	—	ADIE	RC1IE	TX1IE	SSPIE	CCP1IE	TMR2IE	TMR1IE	61
IPR1	—	ADIP	RC1IP	TX1IP	SSPIP	CCP1IP	TMR2IP	TMR1IP	61
RCSTA1	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	61
RCREG1	EUSART1 接收寄存器								61
TXSTA1	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	61
BAUDCON1	ABDOVF	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	62
SPBRGH1	EUSART1 波特率发生器寄存器的高字节								62
SPBRG1	EUSART1 波特率发生器寄存器的低字节								61

图注： — = 未用位，读为 0。同步主控接收不使用阴影单元。

16.4 EUSART 同步从动模式

将 CSRC（TXSTA<7>）清零可进入同步从动模式。此模式与同步主控模式的区别在于移位时钟由CK1引脚上的外部时钟提供 （主控模式中由内部时钟提供）。这使得器件能在任何低功耗模式下发送或接收数据。

16.4.1 EUSART 同步从动发送

除了休眠模式以外，同步主控、从动模式的工作原理是完全相同的。

如果向缓冲器 TXREG1 写入 2 个字，然后执行 SLEEP 指令，则将发生以下事件：

a） 第一个字立即传送到 TSR 寄存器进行发送。

b) 第二个字仍保留在 TXREG1 寄存器中。

c) 不会将标志位 TX1IF 置 1。

d) 当第一个字移出 TSR 后，TXREG1 寄存器将把第二个字传送给 TSR，同时将标志位 TX1IF 置 1。

e) 如果中断使能位 TX1IE 已置 1，中断将把器件从休眠状态唤醒。如果使能了全局中断，程序则会跳转到中断向量处执行。

设置同步从动发送的步骤如下：

1. 通过将 SYNC 和 SPEN 位置 1、CSRC 位清零，使能同步从动串口。
2. 将 CREN 和 SREN 位清零。
3. 若需要中断，将中断使能位 TX1IE 置 1。
4. 若需要发送 9 位数据，将 TX9 置 1。
5. 将使能位 TXEN 位置 1 使能发送。
6. 如果选择发送9位数据，将第9位数据装入TX9D位。
7. 将数据装入 TXREG1 寄存器，启动发送。
8. 若想使用中断，确保将 INTCON 寄存器中的 GIE 和 PEIE 位 （INTCON<7:6>）置 1。

表 16-9： 与同步从动发送相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
PIR1	—	ADIF	RC1IF	TX1IF	SSPIF	CCP1IF	TMR2IF	TMR1IF	61
PIE1	—	ADIE	RC1IE	TX1IE	SSPIE	CCP1IE	TMR2IE	TMR1IE	61
IPR1	—	ADIP	RC1IP	TX1IP	SSPIP	CCP1IP	TMR2IP	TMR1IP	61
RCSTA1	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	61
TXREG1	EUSART1 发送寄存器								61
TXSTA1	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	61
BAUDCON1	ABDOVF	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	62
SPBRGH1	EUSART1 波特率发生器寄存器的高字节								62
SPBRG1	EUSART1 波特率发生器寄存器的低字节								61

图注： — = 未用位，读为 0。同步从动发送不使用阴影单元。

16.4.2 EUSART 同步从动接收

除了休眠模式、空闲模式以及在从动模式下忽略 SREN 位以外，同步主控和从动模式的工作原理完全相同。

如果在进入休眠或空闲模式前将 CREN 位置 1 使能接收，那么在低功耗模式下可以接收到一个数据字。接收到该字后，RSR 寄存器将把数据发送到 RCREG1 寄存器，如果中断使能位 RC1IE 已置 1，产生的中断将把芯片从低功耗模式唤醒。如果使能了全局中断，程序则会跳转到中断向量处执行。

设置同步从动接收操作的步骤如下：

1. 通过将 SYNC 和 SPEN 位置 1、CSRC 位清零使能同步从动串口。
2. 若需要中断，将中断使能位 RC1IE 置 1。
3. 若需要接收 9 位数据，将 RX9 位置 1。
4. 将使能位 CREN 置 1，使能接收。
5. 当接收完成时， RC1IF 位将被置 1。如果中断使能位 RC1IE 已置 1，还将产生一个中断。
6. 读 RCSTA1 寄存器以获取第 9 位数据 （如果使能的话）并判断在接收过程中是否发生了错误。
7. 通过读 RCREG1 寄存器来读取接收到的 8 位数据。
8. 如果发生错误，将 CREN 位清零以清除错误。
9. 若想使用中断，请确保将 INTCON 寄存器中的 GIE 和 PEIE 位 （INTCON<7:6>）置 1。

表 16-10： 与同步从动接收相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
PIR1	—	ADIF	RC1IF	TX1IF	SSPIF	CCP1IF	TMR2IF	TMR1IF	61
PIE1	—	ADIE	RC1IE	TX1IE	SSPIE	CCP1IE	TMR2IE	TMR1IE	61
IPR1	—	ADIP	RC1IP	TX1IP	SSPIP	CCP1IP	TMR2IP	TMR1IP	61
RCSTA1	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	61
RCREG1	EUSART1 接收寄存器								61
TXSTA1	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	61
BAUDCON1	ABDOVF	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	62
SPBRGH1	EUSART1 波特率发生器寄存器的高字节								62
SPBRG1	EUSART1 波特率发生器寄存器的低字节								61

图注： — = 未用位，读为 0。同步从动接收不使用阴影单元。

17.0 可寻址的通用同步 / 异步收发器（AUSART）

可寻址的通用同步 / 异步收发器（Addressable Universal Synchronous Asynchronous Receiver Transmitter，AUSART）在功能上与上一章讨论的增强型 USART 模块非常相似。AUSART为那些不需要自动波特率检测或LIN 总线支持的外部器件串行通信提供了额外的通道。

AUSART 可配置为以下几种工作模式：

- 全双工异步模式
- 半双工同步主控模式
- 半双工同步从动模式

AUSART 模块的引脚与 PORTG（分别为 RG1/TX2/CK2/SEG29 和 RG2/RX2/DT2/SEG28 引脚）的功能复用。要把这些引脚配置为 AUSART，需要具备以下条件：

- SPEN （RCSTA2<7>）位必须被置 1
- TRISG<2> 位必须被置 1
- TRISG<1> 位必须清 0，配置为异步 / 同步主控模式
- TRISG<1> 位必须被置 1，配置为同步从动模式

注：	AUSART 控制根据需要会自动将引脚从输入重新配置为输出。

通过 TXSTA2 和 RXSTA2 这两个寄存器控制可寻址 USART 模块的操作。寄存器 17-1 和寄存器 17-2 分别对这两个寄存器做了详细说明。

寄存器 17-1： TXSTA2：AUSART 发送状态和控制寄存器

R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0	R-1	R/W-0
CSRC	TX9	TXEN(1)	SYNC	—	BRGH	TRMT	TX9D
bit 7							bit 0

bit 7 **CSRC**：时钟源选择位

异步模式：
忽略。

同步模式：
1 = 主控模式 （时钟来自内部 BRG）
0 = 从动模式 （时钟来自外部时钟源）

bit 6 **TX9**：9 位发送使能位
1 = 选择 9 位发送
0 = 选择 8 位发送

bit 5 **TXEN**：发送使能位 (1)
1 = 使能发送
0 = 禁止发送

注 1：同步模式下 SREN/CREN 比 TXEN 优先级高。

bit 4 **SYNC**：AUSART 模式选择位
1 = 同步模式
0 = 异步模式

bit 3 **未用位**：读为 0

bit 2 **BRGH**：高波特率选择位

异步模式：
1 = 高速度
0 = 低速度

同步模式：
在此模式下未使用。

bit 1 **TRMT**：发送移位寄存器状态位
1 = TSR 空
0 = TSR 满

bit 0 **TX9D**：发送数据的第 9 位
此位可以是地址 / 数据位或奇偶校验位。

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

寄存器 17-2： RCSTA2：AUSART 接收状态和控制寄存器

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-0	R-0	R-x
SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D
bit 7							bit 0

bit 7 **SPEN**：串口使能位

1 = 使能串口 （将 RX/DT 和 TX/CK 引脚配置为串口引脚）
0 = 禁止串口 （保持为复位状态）

bit 6 **RX9**：9 位接收使能位

1 = 选择 9 位接收
0 = 选择 8 位接收

bit 5 **SREN**：单字接收使能位

异步模式：
忽略。
同步主控模式：
1 = 使能单字接收
0 = 禁止单字接收
在接收完成后清零该位。
同步从动模式：
忽略。

bit 4 **CREN**：连续接收使能位

异步模式：
1 = 使能接收器
0 = 禁止接收器
同步模式：
1 = 使能连续接收，直到使能位 CREN 被清零 （CREN 比 SREN 优先级高）
0 = 禁止连续接收

bit 3 **ADDEN**：地址检测使能位

9 位异步模式 （RX9 = 1）：
1 = 当 RSR<8> 置 1 时，使能地址检测、中断和装载接收缓冲器
0 = 禁止地址检测、接收所有字节并且第 9 位可作为奇偶校验位
9 位异步模式 （RX9 = 0）：
忽略。

bit2 **FERR**：帧错误位

1 = 帧错误 （读 RCREG 寄存器可更新该位并接收下一个有效字节）
0 = 无帧错误

bit1 **OERR**：溢出错误位

1 = 溢出错误 （清零 CREN 位可将该位清零）
0 = 无溢出错误

bit0 **RX9D**：接收数据的第 9 位

此位可以是地址 / 数据位或奇偶校验位，且必须由用户固件计算得到。

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

17.1 AUSART 波特率发生器 （BRG）

BRG 是一个专用的 8 位发生器，支持 AUSART 的异步和同步模式。

SPBRG2 寄存器控制独立运行的定时器的周期。在异步模式下， BRGH （TXSTA<2>）位还用于控制波特率。在同步模式下可忽略 BRGH 位。表 17-1 给出了主控模式（内部时钟）下，不同 USART 模式的波特率计算公式。

在给定目标波特率和 FOSC 的情况下，可以使用表 17-1 中的公式计算与SPBRG2寄存器中数值最接近的整数。由此可确定波特率的误差。例 17-1 给出了一个计算波特率的示例。表 17-2 显示了不同异步模式下典型的波特率和误差值。使用高波特率 （BRGH = 1）有利于减少波特率误差，或在快速振荡频率条件下获得低波特率。

向 SPBRG2 寄存器写入新值会复位（或清零）BRG 定时器。这可以确保BRG 无需等待定时器溢出就可以输出新的波特率。

17.1.1 在功耗管理模式下工作

器件时钟用于生成所需的波特率。当进入一种功耗管理模式时，新时钟源可能会工作在一个不同的频率下。这可能需要调整 SPBRG2 寄存器中的值。

17.1.2 采样

择多检测电路对 RX2 引脚采样三次，以判定 RX2 引脚上出现的是高电平还是低电平。

表 17-1： 波特率公式

配置位		BRG/AUSART 模式	波特率公式
SYNC	BRGH		
0	0	异步	FOSC/[64 (n + 1)]
0	1	异步	FOSC/[16 (n + 1)]
1	x	同步	FOSC/[4 (n + 1)]

图注： x = 任意值， n = SPBRG2 寄存器的值

例 17-1： 计算波特率误差

器件工作在：FOSC = 16 MHz，目标波特率 = 9600 bps，异步模式， BRGH = 0：
目标波特率 = FOSC/(64 ([SPBRG2] + 1))
求解 SPBRG2：
X = ((FOSC/ 目标波特率)/64) – 1
= ((16000000/9600)/64) – 1
= [25.042] = 25
波特率计算结果 = 16000000/(64 (25 + 1))
= 9615
误差 = (波特率计算结果 – 目标波特率) / 目标波特率
= (9615 – 9600)/9600 = 0.16% =(9615 – 9600)/9600 = 0.16%

表 17-2： 与波特率发生器相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在的页
TXSTA2	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	63
RCSTA2	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	63
SPBRG2	AUSART2 波特率发生器寄存器								63

图注： BRG 未使用阴影单元。

 初稿

表 **17-3**： 异步模式下的波特率

目标波特率（Kbps）	BRGH = 0											
	Fosc = 40.000 MHz			Fosc = 20.000 MHz			Fosc = 10.000 MHz			Fosc = 8.000 MHz		
	实际波特率（Kbps）	误差 %	SPBRG 值（10 进制）	实际波特率（Kbps）	误差 %	SPBRG 值（10 进制）	实际波特率（Kbps）	误差 %	SPBRG 值（10 进制）	实际波特率（Kbps）	误差 %	SPBRG 值（10 进制）
0.3	—	—	—	—	—	—	—	—	—	—	—	—
1.2	—	—	—	1.221	1.73	255	1.202	0.16	129	1.201	-0.16	103
2.4	2.441	1.73	255	2.404	0.16	129	2.404	0.16	64	2.403	-0.16	51
9.6	9.615	0.16	64	9.766	1.73	31	9.766	1.73	15	9.615	-0.16	12
19.2	19.531	1.73	31	19.531	1.73	15	19.531	1.73	7	—	—	—
57.6	56.818	-1.36	10	62.500	8.51	4	52.083	-9.58	2	—	—	—
115.2	125.000	8.51	4	104.167	-9.58	2	78.125	-32.18	1	—	—	—

目标波特率（Kbps）	BRGH = 0								
	Fosc = 4.000 MHz			Fosc = 2.000 MHz			Fosc = 1.000 MHz		
	实际波特率（Kbps）	误差 %	SPBRG 值（10 进制）	实际波特率（Kbps）	误差 %	SPBRG 值（10 进制）	实际波特率（Kbps）	误差 %	SPBRG 值（10 进制）
0.3	0.300	0.16	207	0.300	-0.16	103	0.300	-0.16	51
1.2	1.202	0.16	51	1.201	-0.16	25	1.201	-0.16	12
2.4	2.404	0.16	25	2.403	-0.16	12	—	—	—
9.6	8.929	-6.99	6	—	—	—	—	—	—
19.2	20.833	8.51	2	—	—	—	—	—	—
57.6	62.500	8.51	0	—	—	—	—	—	—
115.2	62.500	-45.75	0	—	—	—	—	—	—

目标波特率（Kbps）	BRGH = 1											
	Fosc = 40.000 MHz			Fosc = 20.000 MHz			Fosc = 10.000 MHz			Fosc = 8.000 MHz		
	实际波特率（Kbps）	误差 %	SPBRG 值（10 进制）	实际波特率（Kbps）	误差 %	SPBRG 值（10 进制）	实际波特率（Kbps）	误差 %	SPBRG 值（10 进制）	实际波特率（Kbps）	误差 %	SPBRG 值（10 进制）
0.3	—	—	—	—	—	—	—	—	—	—	—	—
1.2	—	—	—	—	—	—	—	—	—	—	—	—
2.4	—	—	—	—	—	—	2.441	1.73	255	2.403	-0.16	207
9.6	9.766	1.73	255	9.615	0.16	129	9.615	0.16	64	9.615	-0.16	51
19.2	19.231	0.16	129	19.231	0.16	64	19.531	1.73	31	19.230	-0.16	25
57.6	58.140	0.94	42	56.818	-1.36	21	56.818	-1.36	10	55.555	3.55	8
115.2	113.636	-1.36	21	113.636	-1.36	10	125.000	8.51	4	—	—	—

目标波特率（Kbps）	BRGH = 1								
	Fosc = 4.000 MHz			Fosc = 2.000 MHz			Fosc = 1.000 MHz		
	实际波特率（Kbps）	误差 %	SPBRG 值（10 进制）	实际波特率（Kbps）	误差 %	SPBRG 值（10 进制）	实际波特率（Kbps）	误差 %	SPBRG 值（10 进制）
0.3	—	—	—	—	—	—	0.300	-0.16	207
1.2	1.202	0.16	207	1.201	-0.16	103	1.201	-0.16	51
2.4	2.404	0.16	103	2.403	-0.16	51	2.403	-0.16	25
9.6	9.615	0.16	25	9.615	-0.16	12	—	—	—
19.2	19.231	0.16	12	—	—	—	—	—	—
57.6	62.500	8.51	3	—	—	—	—	—	—
115.2	125.000	8.51	1	—	—	—	—	—	—

17.2 AUSART 异步模式

通过清零 SYNC 位（TXSTA2<4>）选择异步工作模式。在此模式下，AUSART 使用标准的不归零（NRZ）格式（1 个起始位，8 个或 9 个数据位，1 个停止位）。最常用的数据格式为 8 位。片上专用的 8 位波特率发生器可用于产生标准的波特率频率。

AUSAR 首先发送和接收最低有效位。AUSART 的发送器和接收器在功能上是独立的，但采用相同的数据格式和波特率。波特率发生器可以根据 BRGH 位（TXSTA2<2>）的设置产生两种不同的波特率时钟，速率分别为移位速率的 16 倍或 64 倍。硬件不支持奇偶校验，但可以用软件实现，校验位保存在第9个数据位中。

当工作在异步模式下时，AUSART 模块包括以下重要组成部分：

- 波特率发生器
- 采样电路
- 异步发送器
- 异步接收器

17.2.1 AUSART 异步发送器

图 17-1 为 AUSART 发送器的原理框图。发送器的核心是发送（串行）移位寄存器（TSR）。该移位寄存器从读 / 写发送缓冲寄存器 TXREG2 中获取数据。TXREG2 寄存器中的数据由软件装入。在前一次装入数据的停止位发送完成前，不会向 TSR 寄存器装入数据。一旦停止位发送完毕， TXREG2 寄存器中的新数据 （如果有的话）就会被装入 TSR。

一旦 TXREG2 寄存器向 TSR 寄存器传输了数据 （在 1 个 TCY 内发生），TXREG2 寄存器就为空，同时标志位 TX2IF （PIR3<4>）被置 1。可以通过将中断使能位 TX2IE （PIE3<4>）置 1 或清零来使能 / 禁止该中断。只要条件符合，TX2IF 就会置 1，而不管 TX2IE 的状态如何。TX2IF 不能用软件清零，也不会在装入 TXREG2 时立即置 1，而是在装入指令后的第二个指令周期置 1。因此在装入 TXREG2 后立即查询 TX2IF 会得到无效结果。

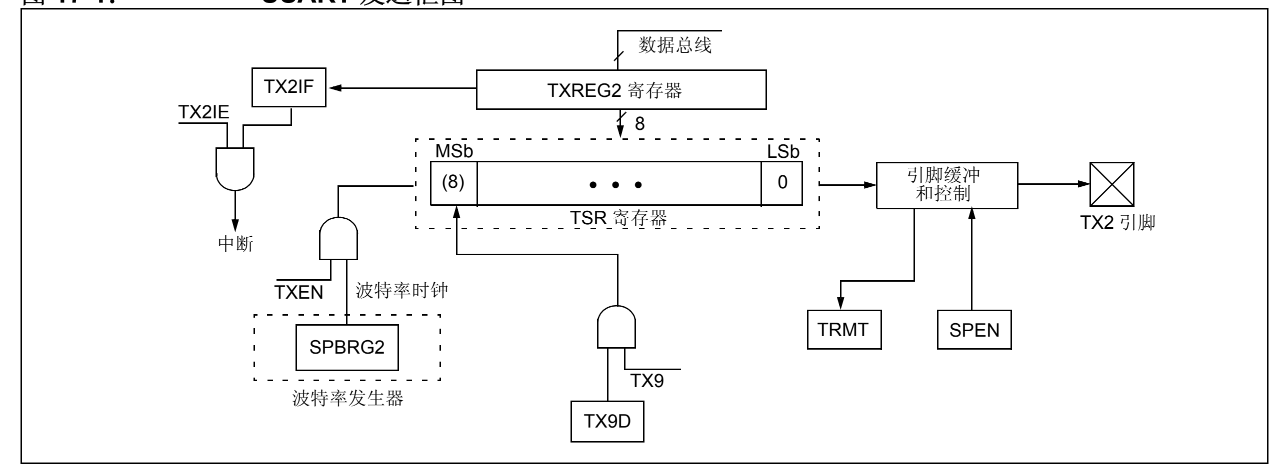
TX2IF 表示的是 TXREG2 寄存器的状态，而另一个位 TRMT （TXSTA2<1>）则表示 TSR 寄存器的状态。TRMT 是只读位，它在 TSR 寄存器为空时置 1。TRMT 位与任何中断均无关联，因此要确定TSR寄存器是否为空，用户只能对此位进行轮询。

注	
1：	TSR 寄存器并未映射到数据存储器中，因此用户不能直接访问它。
2：	当使能位 TXEN 置 1 时，标志位 TX2IF 也会置 1。

设置异步发送的步骤如下：

1. 对 SPBRG2 寄存器进行初始化，设置合适的波特率。按需要将 BRGH 位置 1 或清零，以获得所需的波特率。
2. 将 SYNC 位清零、SPEN 位置 1，使能异步串口。
3. 若需要中断，将中断使能位 TX2IE 置 1。
4. 若需要发送 9 位数据，将发送位 TX9 置 1。发送的第 9 位可以是地址位也可以是数据位。
5. 通过将 TXEN 位置 1 使能发送，此操作同时也会使 TX2IF 位置 1。
6. 如果选择发送 9 位数据，应该将第 9 位数据装入 TX9D 位。
7. 将数据装入 TXREG2 寄存器 （启动发送）。
8. 若想使用中断，确保将 INTCON 寄存器中的 GIE 和 PEIE 位 （INTCON<7:6>）置 1。


图 17-1：　USART 发送框图



　初稿　

图 17-2： 异步发送时序

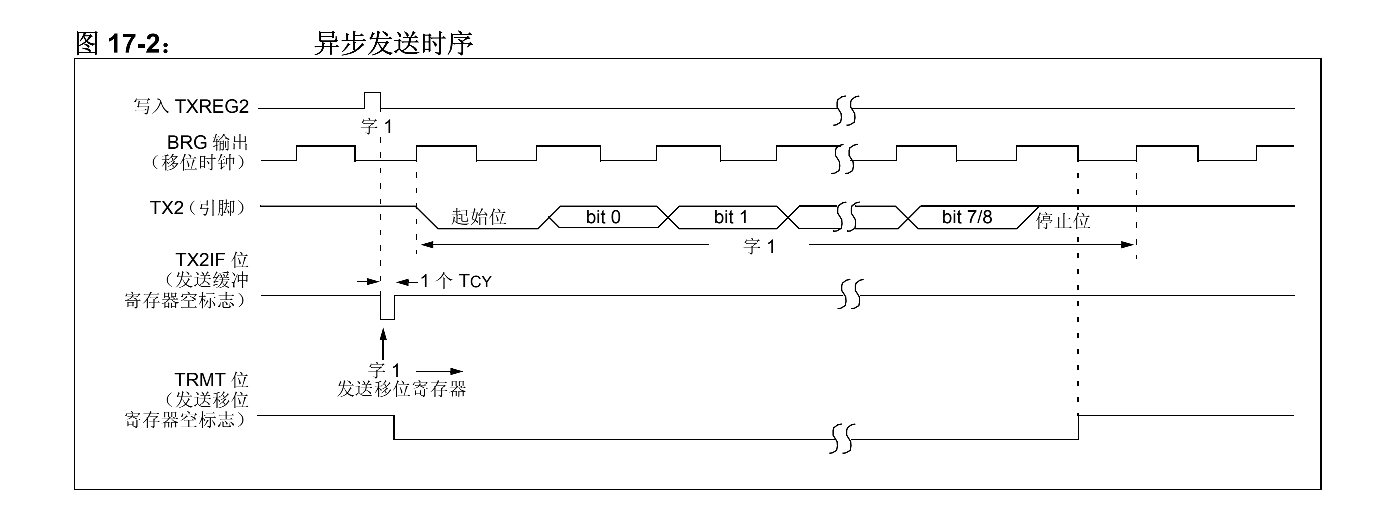
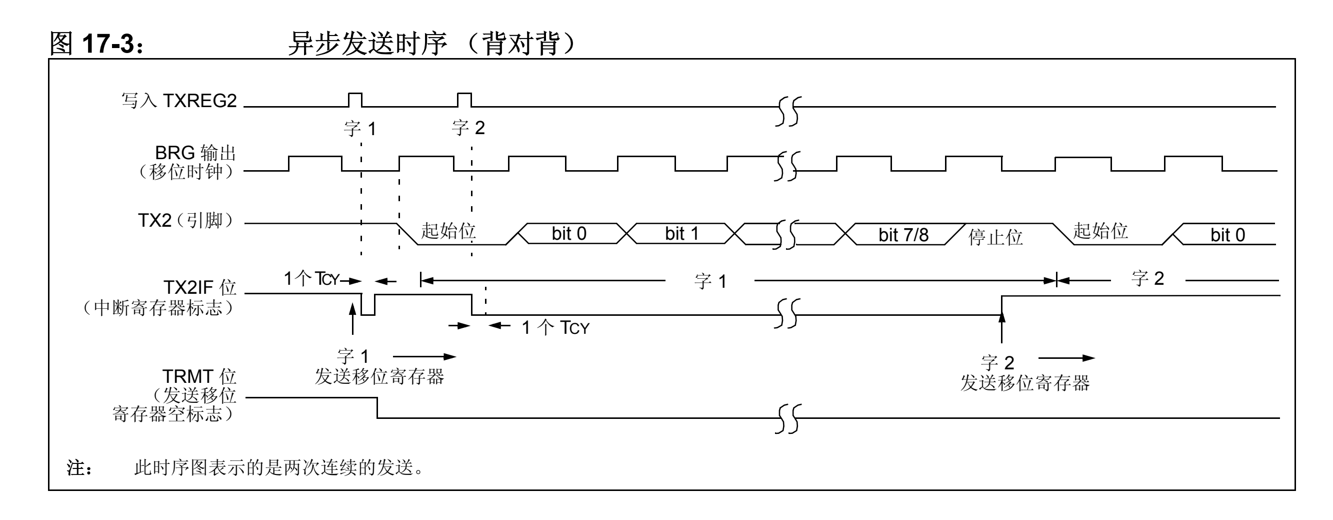


图 17-3： 异步发送时序 （背对背）



注： 此时序图表示的是两次连续的发送。

表 17-4： 与异步发送相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在的页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
PIR3	—	LCDIF	RC2IF	TX2IF	—	—	—	—	61
PIE3	—	LCDIE	RC2IE	TX2IE	—	—	—	—	61
IPR3	—	LCDIP	RC2IP	TX2IP	—	—	—	—	61
RCSTA2	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	63
TXREG2	AUSART2 发送寄存器								63
TXSTA2	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	63
SPBRG2	AUSART2 波特率发生器寄存器								63

图注： — = 未用位，读为 0。异步发送不使用阴影单元。

17.2.2 AUSART 异步接收器

图 17-4 显示了接收器的原理框图。在 RX2 引脚上接收数据，并驱动数据恢复电路。数据恢复电路实际上是一个 16 倍波特率的高速移位器，而主接收串行移位寄存器则工作在此比特率下或 FOSC 下。此模式通常用于 RS-232 系统。

设置异步接收的步骤如下：

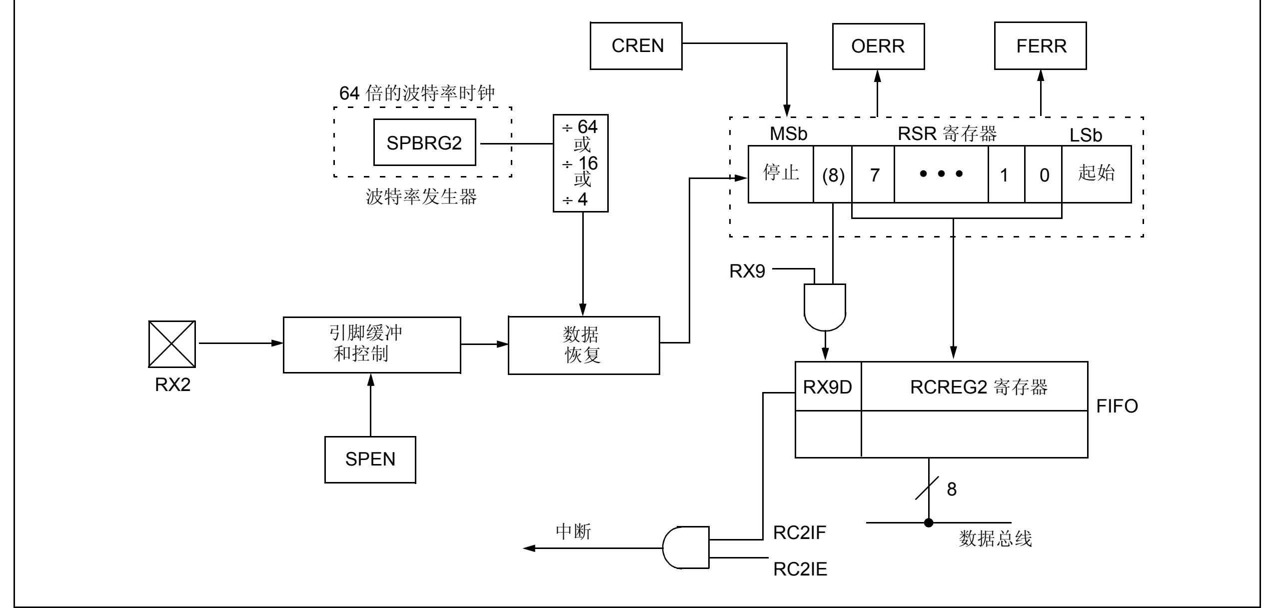
1. 对 SPBRG2 寄存器进行初始化，设置合适的波特率。按需要将 BRGH 位置 1 或清零，以获得所需的波特率。
2. 将 SYNC 位清零、SPEN 位置 1 使能异步串口。
3. 若需要中断，将中断使能位 RC2IE 置 1。
4. 若需要接收 9 位数据，将 RX9 位置 1。
5. 将 CREN 位置 1 使能接收。
6. 当接收完成时标志位 RC2IF 将被置 1，此时如果中断使能位 RC2IE 已置 1，还将产生一个中断。
7. 读 RCSTA2 寄存器获取第 9 位数据 （如果已使能），并判断在接收过程中是否发生了错误。
8. 通过读 RCREG2 寄存器来读取接收到的 8 位数据。
9. 如果发生错误，通过将使能位 CREN 清零来清除错误。
10. 若想使用中断，请确保将 INTCON 寄存器中的 GIE 和 PEIE 位 （INTCON<7:6>）置 1。

17.2.3 设置带有地址检测功能的 9 位模式

此模式通常用在 RS-485 系统中。按如下步骤设置使能地址检测的异步接收：

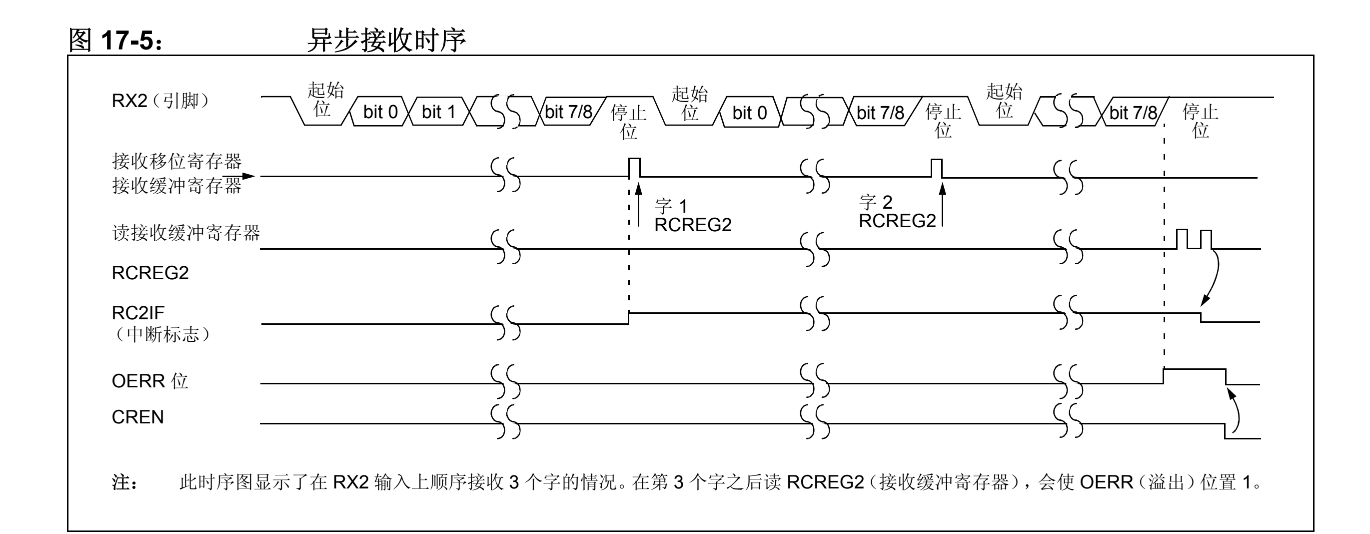
1. 对 SPBRG2 寄存器进行初始化，设置合适的波特率。按需要将 BRGH 和 BRG16 位置 1 或清零，以获得所需的波特率。
2. 将 SYNC 位清零、SPEN 位置 1，使能异步串口。
3. 若需要中断，请将 RCEN 位置 1 并使用 RC2IP 位选择所需的优先级。
4. 将 RX9 位置 1 使能 9 位接收。
5. 将 ADDEN 位置 1 使能地址检测。
6. 将 CREN 位置 1 使能接收。
7. 当接收完成时 RC2IF 位将被置 1。此时如果 RC2IE 和 GIE 位已置 1，还将响应中断。
8. 读 RCSTA2 寄存器判断在接收时是否发生了错误，同时读取第 9 位数据 （如果适用）。
9. 读 RCREG2 以判断是否正在对器件进行寻址。
10. 如果发生错误，将 CREN 位清零。
11. 如果已经找到器件，将 ADDEN 位清零以允许接收到的数据进入接收缓冲器，并中断 CPU。

图 17-4： AUSART 接收框图



 初稿

图 17-5： 异步接收时序



注： 此时序图显示了在 RX2 输入上顺序接收 3 个字的情况。在第 3 个字之后读 RCREG2（接收缓冲寄存器），会使 OERR（溢出）位置 1。

表 17-5： 与异步接收相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
PIR3	—	LCDIF	RC2IF	TX2IF	—	—	—	—	61
PIE3	—	LCDIE	RC2IE	TX2IE	—	—	—	—	61
IPR3	—	LCDIP	RC2IP	TX2IP	—	—	—	—	61
RCSTA2	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	63
RCREG2	AUSART2 接收寄存器								63
TXSTA2	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	63
SPBRG2	AUSART2 波特率发生器寄存器								63

图注： — = 未用位，读为 0。异步接收不使用阴影单元。

17.3 AUSART 同步主控模式

将 CSRC 位 （TXSTA2<7>）置 1 可以进入同步主控模式。在此模式中，数据以半双工方式发送（即发送和接收不同时进行） 。发送数据时，禁止接收，反之亦然。将 SYNC 位 （TXSTA2<4>）置 1 可进入同步模式。此外，应将使能位 SPEN（RCSTA2<7>）置 1，以把 TX2 和RX2引脚分别配置为CK2（时钟）和DT2（数据）线。

主控模式意味着处理器在CK2时钟线上发送主控时钟信号。

17.3.1 AUSART 同步主控发送

图 17-1 显示了 AUSART 发送器的原理框图。发送器的核心是发送（串行）移位寄存器（TSR）。该移位寄存器从读 / 写发送缓冲寄存器 TXREG2 中获取数据。TXREG2 寄存器中的数据由软件装入。在前一次装入数据的最后一位发送完成前，不会向TSR寄存器装入新数据。一旦最后一位发送完成，就会将 TXREG2 寄存器中的新数据（如果有的话）装入 TSR。

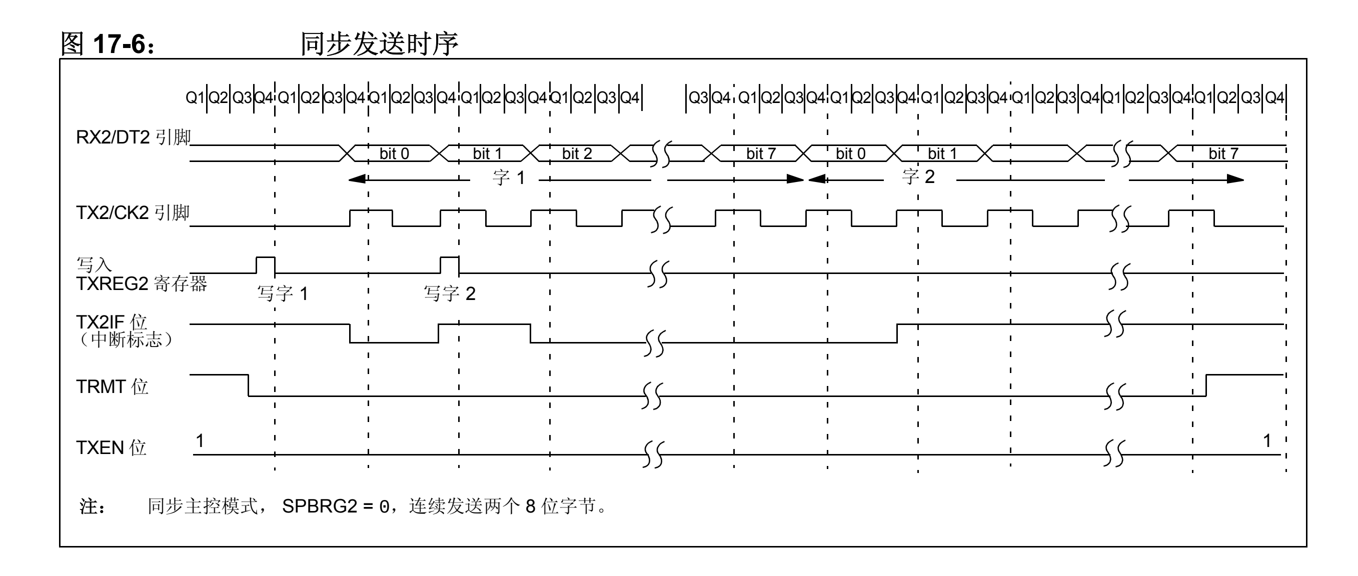
一旦 TXREG2 寄存器向 TSR 寄存器传输了数据 （在 1 个 TCY 内发生），TXREG2 寄存器就为空，同时标志位 TX2IF（PIR3<4>）置 1。可以通过将中断使能位 TX2IE（PIE3<4>）置 1 或清零来使能 / 禁止该中断。只要条件符合， TX2IF 就会置 1，而不管 TX2IE 的状态如何。TX2IF 不能用软件清零。只有把新数据装入 TXREG2 寄存器时 TX2IF 位才会复位。

标志位 TX2IF 表示的是 TXREG2 寄存器的状态，而另一个位 TRMT（TXSTA2<1>）则表示 TSR 寄存器的状态。TRMT 位是一个只读位，当 TSR 为空时， TRMT 置 1。TRMT 位与任何中断逻辑均无关联，因此要判断 TSR 寄存器是否为空，用户只能对该位进行轮询。TSR 并未映射到数据存储器中，所以用户不能直接访问它。

设置同步主控发送的步骤如下：

1. 对 SPBRG2 寄存器进行初始化 , 设置合适的波特率。
2. 将 SYNC、SPEN 和 CSRC 位置 1，使能同步主控串口。
3. 若需要中断，将中断使能位 TX2IE 置 1。
4. 若需要发送 9 位数据，将 TX9 置 1。
5. 将 TXEN 位置 1 使能发送。
6. 如果选择发送 9 位数据，应该将第 9 位数据装入 TX9D 位。
7. 将数据装入 TXREG2 寄存器，以启动发送。
8. 若想使用中断，确保将 INTCON 寄存器中的 GIE 和 PEIE 位 （INTCON<7:6>）置 1。

图 17-6： 同步发送时序



注： 同步主控模式， SPBRG2 = 0，连续发送两个 8 位字节。

图 **17-7**： 同步发送时序 （由 **TXEN** 位控制）

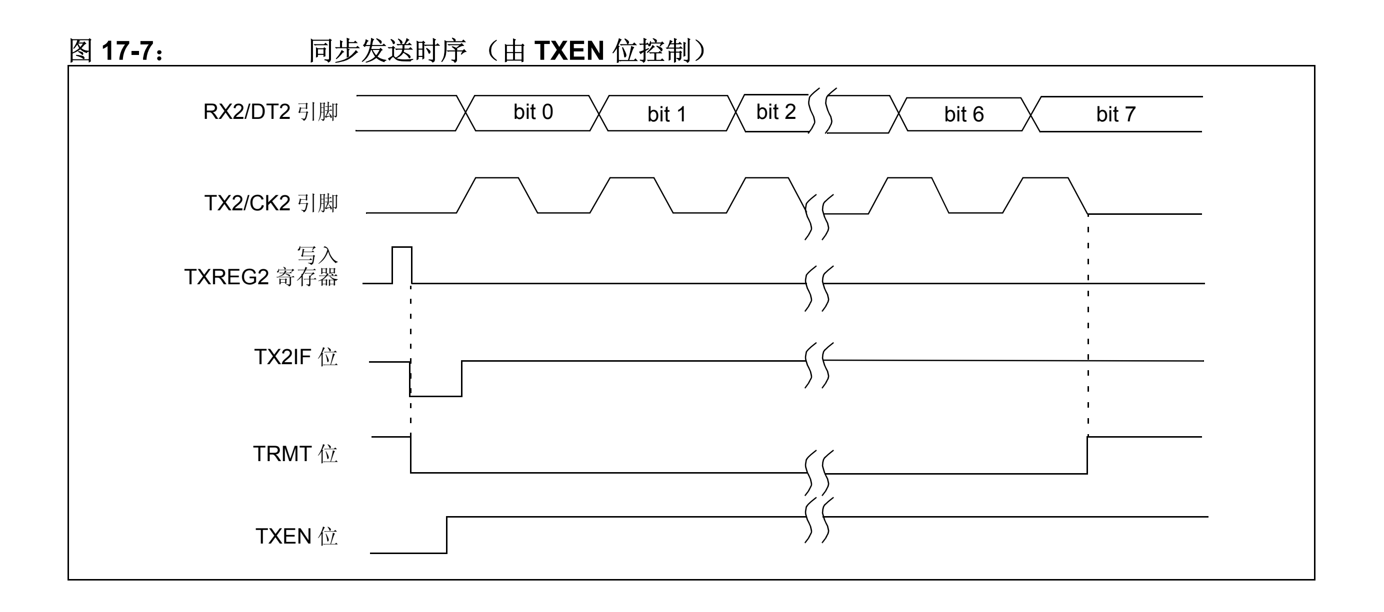


表 **17-6**： 与同步主控发送相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
PIR3	—	LCDIF	RC2IF	TX2IF	—	—	—	—	61
PIE3	—	LCDIE	RC2IE	TX2IE	—	—	—	—	61
IPR3	—	LCDIP	RC2IP	TX2IP	—	—	—	—	61
RCSTA2	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	63
TXREG2	AUSART2 发送寄存器								63
TXSTA2	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	63
SPBRG2	AUSART2 波特率发生器寄存器								63

图注： — = 未用位，读为 0。同步主控发送不使用阴影单元。

17.3.2 AUSART 同步主控接收

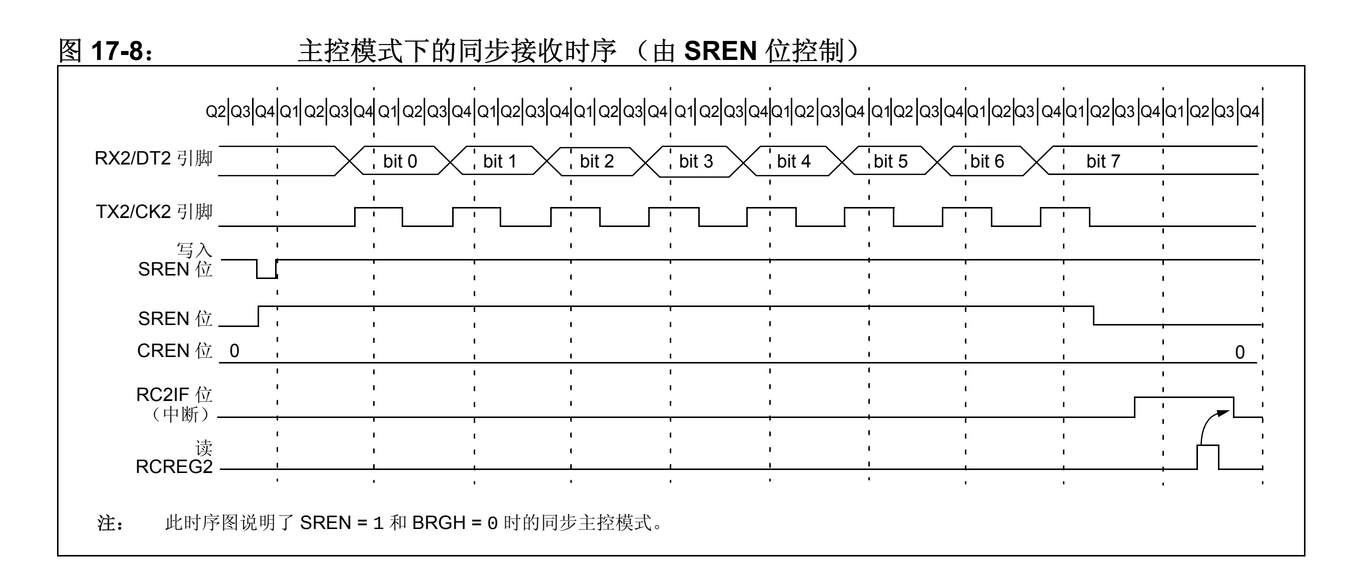
一旦选择了同步模式，只要将单字接收使能位 SREN（RCSTA2<5>）或连续接收使能位 CREN（RCSTA2<4>）置 1，即可使能接收。在时钟的下降沿采样 RX2 引脚上的数据。

如果将使能位 SREN 置 1，则只接收单字；如果将使能位 CREN 置 1，则会连续接收数据，直到将 CREN 位清零。如果两个位均被置 1，则 CREN 具有优先权。

设置同步主控接收的步骤如下：

1. 对 SPBRG2 寄存器进行初始化 , 设置合适的波特率。
2. 将 SYNC、SPEN 和 CSRC 位置 1，使能同步主控串口。
3. 确保将 CREN 和 SREN 位清零。
4. 若需要中断，将中断使能位 RC2IE 置 1。
5. 若需要接收 9 位数据，将 RX9 位置1。
6. 若需要单字接收，将 SREN 位置 1 ；若需要连续接收，将 CREN 位置 1。
7. 当接收完成时中断标志位 RC2IF 将置 1，此时如果中断使能位 RC2IE 已置 1，则还将产生一个中断。
8. 读 RCSTA2 寄存器以获取第 9 位数据 （如果使能的话）并判断在接收过程中是否发生了错误。
9. 通过读 RCREG2 寄存器来读取接收到的 8 位数据。
10. 如果发生错误，将 CREN 位清零以清除错误。
11. 若想使用中断，请确保将 INTCON 寄存器中的 GIE 和 PEIE 位 （INTCON<7:6>）置 1。

图 17-8： 主控模式下的同步接收时序 （由 SREN 位控制）



注： 此时序图说明了 SREN = 1 和 BRGH = 0 时的同步主控模式。

表 17-7： 与同步主控接收相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
PIR3	—	LCDIF	RC2IF	TX2IF	—	—	—	—	61
PIE3	—	LCDIE	RC2IE	TX2IE	—	—	—	—	61
IPR3	—	LCDIP	RC2IP	TX2IP	—	—	—	—	61
RCSTA2	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	63
RCREG2	AUSART2 接收寄存器								63
TXSTA2	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	63
SPBRG2	AUSART2 波特率发生器寄存器的低字节								63

图注： — = 未用位，读为 0。同步主控接收不使用阴影单元。

17.4 AUSART 同步从动模式

将 CSRC （TXSTA2<7>）清零可进入同步从动模式。此模式与同步主控模式的区别在于移位时钟由CK2引脚上的外部时钟提供（在主控模式中由内部时钟源提供）。这使得器件能在低功耗模式下发送或接收数据。

17.4.1 AUSART 同步从动发送

除了休眠模式以外，同步主控、从动模式的工作原理是完全相同的。

如果向缓冲器 TXREG2 写入 2 个字，然后执行 SLEEP 指令，将发生以下事件：

a） 第一个字立即传送到 TSR 寄存器进行发送。
b) 第二个字仍保留在 TXREG2 寄存器中。
c) 标志位 TX2IF 不会被置 1。
d) 当第一个字移出 TSR 后， TXREG2 寄存器把第二个字送入 TSR，同时将标志位 TX2IF 置 1。
e) 如果中断使能位 TX2IE 已置 1，中断将把芯片从休眠状态唤醒。如果使能了全局中断，程序则会跳转到中断向量处执行。

设置同步从动发送得步骤如下：

1. 通过将 SYNC 和 SPEN 位置 1、CSRC 位清零，使能同步从动串口。
2. 将 CREN 和 SREN 位清零。
3. 若需要中断，将中断使能位 TX2IE 置 1。
4. 若需要发送 9 位数据，将 TX9 置 1。
5. 将使能位 TXEN 位置 1 使能发送。
6. 如果选择发送 9 位数据，应该将第 9 位数据装入 TX9D 位。
7. 将数据装入 TXREG2 寄存器启动发送。
8. 若想使用中断，请确保将 INTCON 寄存器中的 GIE 和 PEIE 位 （INTCON<7:6>）置 1。

表 17-8： 与同步从动发送相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
PIR3	—	LCDIF	RC2IF	TX2IF	—	—	—	—	61
PIE3	—	LCDIE	RC2IE	TX2IE	—	—	—	—	61
IPR3	—	LCDIP	RC2IP	TX2IP	—	—	—	—	61
RCSTA2	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	63
TXREG2	AUSART2 发送寄存器								63
TXSTA2	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	63
SPBRG2	AUSART2 波特率发生器寄存器的低字节								63

图注： — = 未用位，读为 0。同步从动发送不使用阴影单元。

17.4.2 AUSART 同步从动接收

除了休眠模式、空闲模式以及在从动模式下忽略 SREN 位以外，同步主控模式和从动模式的工作原理完全相同。

如果在进入休眠或空闲模式前将 CREN 位置 1 使能接收，那么在低功耗模式下可以接收一个数据字。接收到该字后，RSR 寄存器将把数据发送到 RCREG2 寄存器，如果中断使能位 RC2IE 已置 1，产生的中断将把芯片从低功耗模式中唤醒。如果使能了全局中断，程序则会跳转到中断向量处执行。

设置同步从动接收的步骤如下：

1. 通过将 SYNC 和 SPEN 位置 1、CSRC 位清零，使能同步从动串口。
2. 若需要中断，将中断使能位 RC2IE 置 1。
3. 若需要接收 9 位数据，将 RX9 位置 1。
4. 将使能位 CREN 置 1 使能接收。
5. 当接收完成时， RC2IF 位将置 1。如果中断使能位 RC2IE 已置 1，还将产生一个中断。
6. 读 RCSTA2 寄存器以获取第 9 位数据 （如果使能的话）并判断在接收过程中是否发生了错误。
7. 通过读 RCREG2 寄存器来读取接收到的 8 位数据。
8. 如果发生错误，将 CREN 位清零以清除错误。
9. 若想使用中断，请确保将 INTCON 寄存器中的 GIE 和 PEIE 位 （INTCON<7:6>）置 1。

表 17-9： 与同步从动接收相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
PIR3	—	LCDIF	RC2IF	TX2IF	—	—	—	—	61
PIE3	—	LCDIE	RC2IE	TX2IE	—	—	—	—	61
IPR3	—	LCDIP	RC2IP	TX2IP	—	—	—	—	61
RCSTA2	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	63
RCREG2	AUSART2 接收寄存器								63
TXSTA2	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	63
SPBRG2	AUSART2 波特率发生器寄存器的低字节								63

图注： — = 未用位，读为 0。同步从动接收不使用阴影单元。

18.0 10 位模数转换器 （A/D）模块

PIC18F6X90/8X90 器件的模数 （Analog-to-Digital，A/D）转换器模块有 12 路输入。此模块能将一个模拟输入信号转换成相应的 10 位数字信号。

此模块有五个寄存器：

- A/D 转换结果高位寄存器 （ADRESH）
- A/D 转换结果低位寄存器 （ADRESL）
- A/D 转换控制寄存器 0 （ADCON0）
- A/D 转换控制寄存器 1 （ADCON1）
- A/D 转换控制寄存器 2 （ADCON2）

ADCON0 寄存器（如寄存器 18-1 所示）控制 A/D 模块的工作。ADCON1 寄存器（如寄存器 18-2 所示）配置端口引脚功能。 ADCON2 寄存器 （如寄存器 18-3 所示）配置 A/D 时钟源、可编程采样时间和输出结果的对齐方式。

寄存器 18-1： ADCON0：A/D 控制寄存器 0

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	CHS3	CHS2	CHS1	CHS0	GO/$\overline{\text{DONE}}$	ADON
bit 7							bit 0

bit 7-6 未用位：读作 0

bit 5-2 **CHS3:CHS0**：模拟通道选择位

0000 = 通道 0 （AN0）
0001 = 通道 1 （AN1）
0010 = 通道 2 （AN2）
0011 = 通道 3 （AN3）
0100 = 通道 4 （AN4）
0101 = 通道 5 （AN5）
0110 = 通道 6 （AN6）
0111 = 通道 7 （AN7）
1000 = 通道 8 （AN8）
1001 = 通道 9 （AN9）
1010 = 通道 10 （AN10）
1011 = 通道 11 （AN11）
1100 = 未用通道 (1)
1101 = 未用通道 (1)
1110 = 未用通道 (1)
1111 = 未用通道 (1)

注 1：在未用通道上执行转换会返回不确定的输入值。

bit 1 **GO/$\overline{\text{DONE}}$**：A/D 转换状态位

当 ADON =1 时：
1 = A/D 转换正在进行
0 = A/D 空闲

bit 0 **ADON**：A/D 模块使能位

1 = 使能 A/D 转换器模块
0 = 禁止 A/D 转换器模块

图注：			
R = 可读位	W = 可写位	U = 未用位，读作 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

寄存器 18-2： **ADCON1：A/D 控制寄存器 1**

U-0	U-0	R/W-0	R/W-0	R/W-q	R/W-q	R/W-q	R/W-q
—	—	VCFG1	VCFG0	PCFG3	PCFG2	PCFG1	PCFG0
bit 7							bit 0

bit 7-6 未用位：读为 0

bit 5 **VCFG1**：参考电压配置位（VREF 负参考电压源）：
1 = VREF- (AN2)
0 = AVSS

bit 4 **VCFG0**：参考电压配置位（VREF 正参考电压源）：
1 = VREF+ (AN3)
0 = AVDD

bit 3-0 **PCFG3:PCFG0**：A/D 端口配置控制位：

PCFG3:PCFG0	AN11	AN10	AN9	AN8	AN7	AN6	AN5	AN4	AN3	AN2	AN1	AN0
0000	A	A	A	A	A	A	A	A	A	A	A	A
0001	A	A	A	A	A	A	A	A	A	A	A	A
0010	A	A	A	A	A	A	A	A	A	A	A	A
0011	A	A	A	A	A	A	A	A	A	A	A	A
0100	D	A	A	A	A	A	A	A	A	A	A	A
0101	D	D	A	A	A	A	A	A	A	A	A	A
0110	D	D	D	A	A	A	A	A	A	A	A	A
0111	D	D	D	D	A	A	A	A	A	A	A	A
1000	D	D	D	D	D	A	A	A	A	A	A	A
1001	D	D	D	D	D	D	A	A	A	A	A	A
1010	D	D	D	D	D	D	D	A	A	A	A	A
1011	D	D	D	D	D	D	D	D	A	A	A	A
1100	D	D	D	D	D	D	D	D	D	A	A	A
1101	D	D	D	D	D	D	D	D	D	D	A	A
1110	D	D	D	D	D	D	D	D	D	D	D	A
1111	D	D	D	D	D	D	D	D	D	D	D	D

A = 模拟输入　　D = 数字 I/O

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

寄存器 18-3： ADCON2：A/D 控制寄存器 2

R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ADFM	—	ACQT2	ACQT1	ACQT0	ADCS2	ADCS1	ADCS0
bit 7							bit 0

bit 7 **ADFM**：A/D 结果格式选择位
1 = 右对齐
0 = 左对齐

bit 6 未用位：读为 0

bit 5-3 **ACQT2:ACQT0**：A/D 采样时间选择位
111 = 20 个 TAD
110 = 16 个 TAD
101 = 12 个 TAD
100 = 8 个 TAD
011 = 6 个 TAD
010 = 4 个 TAD
001 = 2 个 TAD
000 = 0 个 TAD[1]

bit 2-0 **ADCS2:ADCS0**：A/D 转换时钟选择位
111 = FRC （时钟来自 A/D 模块 RC 振荡器）[1]
110 = FOSC/64
101 = FOSC/16
100 = FOSC/4
011 = FRC （时钟来自 A/D 模块 RC 振荡器）[1]
010 = FOSC/32
001 = FOSC/8
000 = FOSC/2

注 1：如果选择了 FRC 时钟源，在 A/D 时钟启动之前会加上一个 TCY （指令周期）的延迟。这可以保证在开始转换之前执行 SLEEP 指令。

图注：			
R = 可读位	W = 可写位	U = 未用位，读作 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

可通过软件选择器件的正电源电压和负电源电压（AVDD 和 AVSS）或 RA3/AN3/VREF+/SEG17 和 RA2/AN2/VREF-/SEG16 引脚上的电压作为 A/D 转换的模拟参考电压。

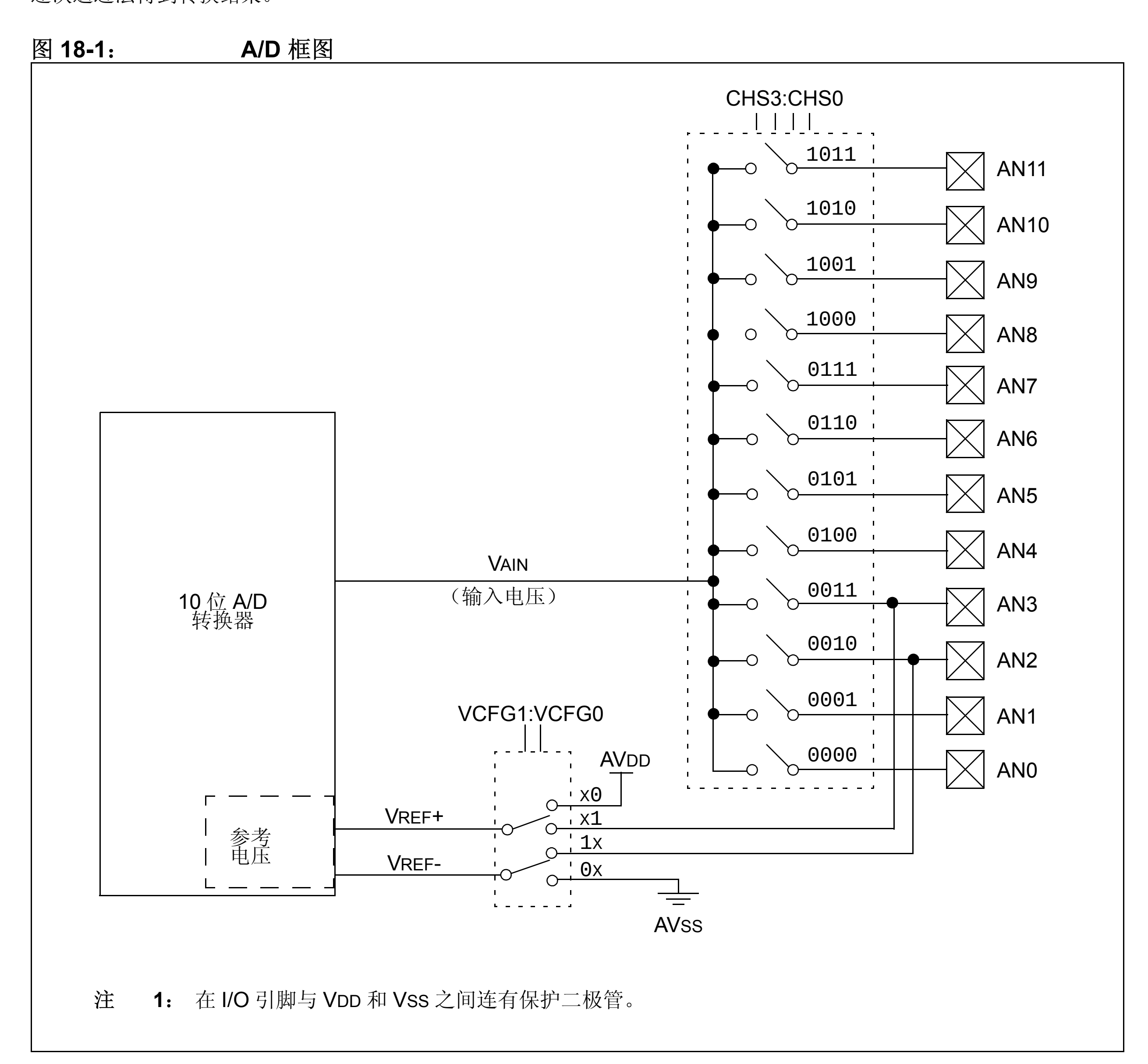
A/D 转换器具有可在休眠状态下工作的特性。要使 A/D 转换器在休眠状态下工作， A/D 转换时钟必须来自于 A/D 模块内部的 RC 振荡器。

采样保持电路的输出是转换器的输入，A/D 转换器采用逐次逼近法得到转换结果。

器件复位操作将强制所有寄存器进入复位状态，这将迫使 A/D 模块关闭并中止正在进行的转换。

与 A/D 转换器相关的每个端口引脚都可以被配置为模拟输入或数字 I/O。ADRESH 和 ADRESL 寄存器保存 A/D 转换的结果。当 A/D 转换完成时，结果被装入 ADRESH/ADRESL寄存器，GO/DONE位（在ADCON0 寄存器中）被清零且 A/D 中断标志位 ADIF 位被置 1。A/D 模块的原理框图见图 18-1。

图 18-1： A/D 框图



注 1： 在 I/O 引脚与 VDD 和 VSS 之间连有保护二极管。

上电复位时，ADRESH:ADRESL 寄存器的值保持不变。上电复位后， ADRESH:ADRESL 寄存器的值不确定。

根据需要配置好 A/D 模块后，在开始转换前必须先对所选择的通道进行采样。模拟输入通道的相应 TRIS 位必须设置为输入。采集时间的确定请参见**第 18.1 节“A/D 采集要求”**。在采样完成之后，即可启动 A/D 转换。采集时间可以被编程位于 GO/DONE 位置 1 和启动转换之间。

在执行 A/D 转换时应该遵循以下步骤：

1. 配置 A/D 模块：
   - 配置模拟引脚、参考电压和数字 I/O （通过 ADCON1 寄存器）
   - 选择 A/D 输入通道 （通过 ADCON0 寄存器）
   - 选择 A/D 采集时间 （通过 ADCON2 寄存器）
   - 选择 A/D 转换时钟 （通过 ADCON2 寄存器）
   - 使能 A/D 模块 （通过 ADCON0 寄存器）
2. 需要时，配置 A/D 中断：
   - ADIF 位清零
   - ADIE 位置 1
   - GIE 位置 1
3. 需要时，等待所需的采样时间。
4. 启动转换：
   - 将 GO/DONE 位置 1（通过 ADCON0 寄存器）
5. 等待 A/D 转换完成，通过以下两种方法之一判断转换是否完成：
   - 轮询 GO/DONE 位是否被清零

   或

   - 等待 A/D 中断
6. 读取 A/D 结果寄存器（ADRESH:ADRESL），需要时将 ADIF 位清零。
7. 如需再次进行 A/D 转换，返回步骤 1 或步骤 2。将每位的 A/D 转换时间定义为 TAD，在下一次采集开始前至少需要等待 3 个 TAD。

图 18-2： A/D 转换方式

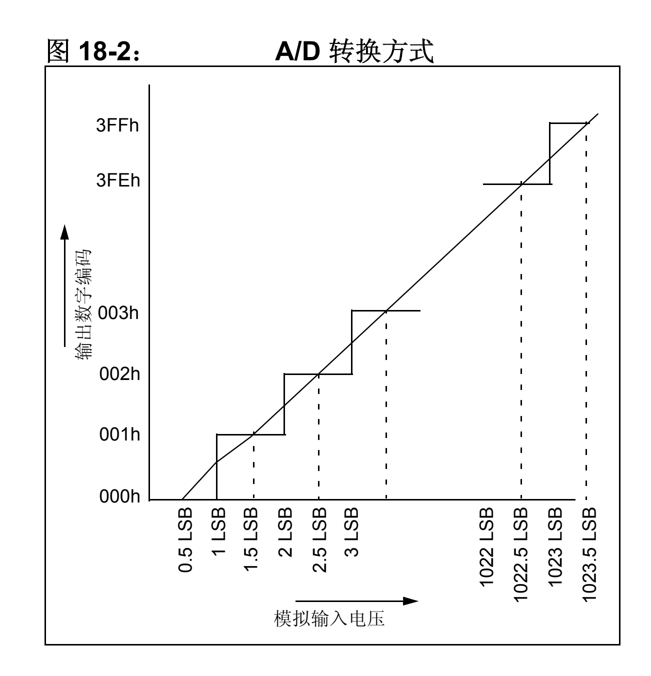
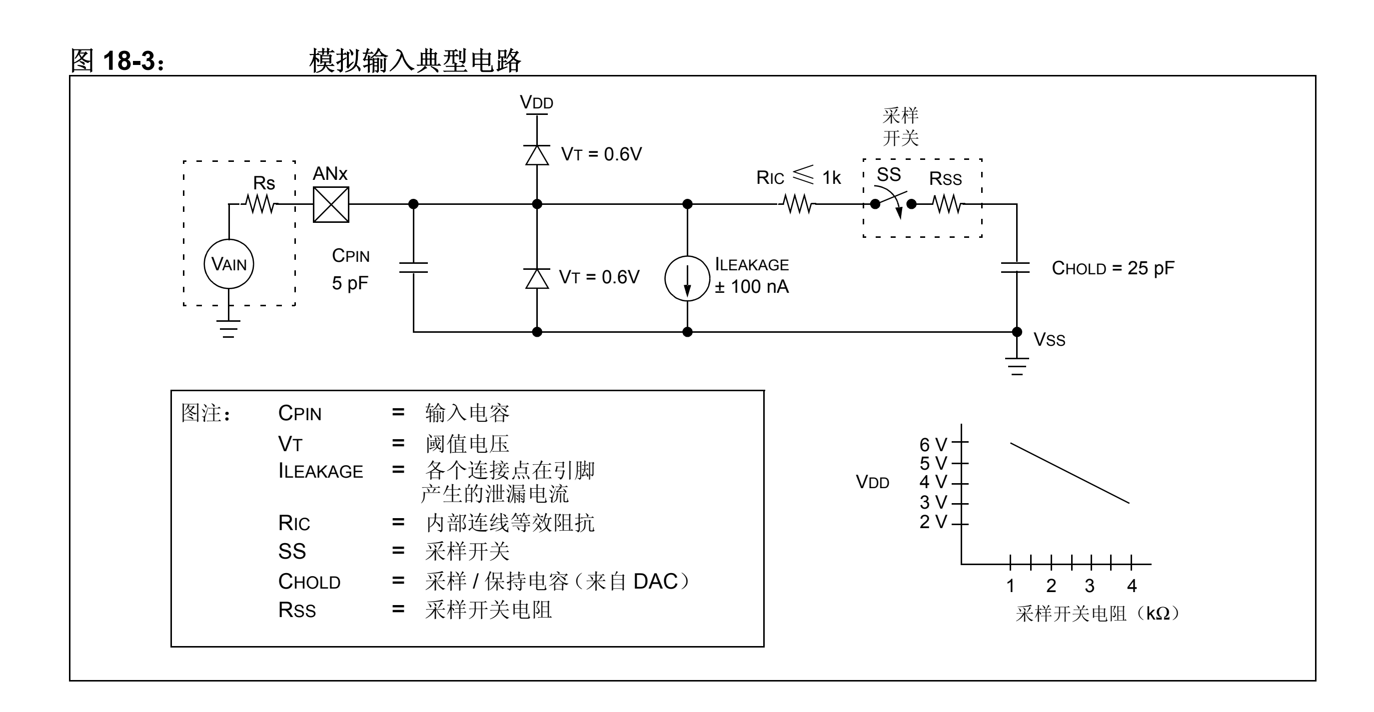


图 18-3： 模拟输入典型电路



18.1 A/D 采集要求

为了使 A/D 转换达到规定精度，必须使充电保持电容（C_{HOLD}）充满至输入通道的电平。图 18-3 给出了模拟输入电路模型。电源阻抗（R_S）和内部采样开关阻抗（R_{SS}）直接影响给电容 C_{HOLD} 充电所需要的时间。采样开关阻抗（R_{SS}）值随器件电压（V_{DD}）不同变化。电源阻抗影响模拟输入的偏置电压（由于引脚泄漏电流的原因）。**模拟信号源的最大阻抗推荐值为 2.5 kΩ**。选择（改变）模拟输入通道后，必须对通道进行采样才能启动转换，采集时间必须大于最小采集时间。

注：	当开始转换时，将保持电容与输入引脚断开。

可以使用公式 18-1 来计算最小采集时间。该公式假设误差为 1/2 LSb（A/D 转换需要 1024 步）， 1/2 LSb 误差是 A/D 达到规定分辨率所允许的最大误差。

例 18-3 显示了所需的最小采集时间 T_{ACQ} 的计算过程。计算结果基于以下假设：

C_{HOLD}	=	25 pF
Rs	=	2.5 kΩ
转换误差	≤	1/2 LSb
V_{DD}	=	5V → Rss = 2 kΩ
温度	=	85°C （系统最大值）

公式 18-1： 采集时间

$$T_{ACQ} = \text{放大器稳定时间} + \text{保持电容充电时间} + \text{温度系数}$$
$$= T_{AMP} + T_C + T_{COFF}$$

公式 18-2： **A/D 最小充电时间**

$$V_{HOLD} = (V_{REF} - (V_{REF}/2048)) \cdot (1 - e^{(-T_C/C_{HOLD}(R_{IC} + R_{SS} + R_S))})$$

或

$$T_C = -(C_{HOLD})(R_{IC} + R_{SS} + R_S) \ln(1/2048)$$

公式 18-3： 计算所需要的最小采集时间

$$T_{ACQ} = T_{AMP} + T_C + T_{COFF}$$

$$T_{AMP} = 0.2\ \mu s$$

$$T_{COFF} = (\text{Temp} - 25°C)(0.02\ \mu s/°C)$$
$$(50°C - 25°C)(0.02\ \mu s/°C)$$
$$1.2\ \mu s$$

只有在温度 >25°C 时才需要温度系数。当温度低于 25°C 时， T_{COFF} = 0 ms。

$$T_C = -(C_{HOLD})(R_{IC} + R_{SS} + R_S) \ln(1/2047)\ \mu s$$
$$-(25\ pF)(1\ k\Omega + 2\ k\Omega + 2.5\ k\Omega) \ln(0.0004883)\ \mu s$$
$$5.03\ \mu s$$

$$T_{ACQ} = 0.2\ \mu s + 5\ \mu s + 1.2\ \mu s$$
$$6.4\ \mu s$$

 初稿

18.2 选择和配置自动采集时间

每当 GO/DONE 位置 1，用户就可利用 ADCON2 寄存器选择采集时间。

当 GO/DONE 位置 1 时，停止采样，开始转换。用户有责任确保在选择输入通道和 GO/DONE 置 1 之间经过了必需的采集时间。当 ACQT2:ACQT0 位（ADCON2<5:3>）保持在复位状态 （000），或器件本身不提供可编程采集时间时，就需要由用户确保必需的采集时间。

如果需要，可将 ACQT 位置 1 为 A/D 模块选择可编程采集时间。当 GO/DONE 位置 1 时，A/D 模块继续对输入进行采样，采样持续的时间是所选择的采集时间，然后自动开始转换。由于采集时间已被编程，因此在选择通道和 GO/DONE 位置 1 之间无需另外等待一个采集时间。

在这两种情况下，当转换完成时， GO/DONE 位均被清零， ADIF 标志位均被置 1 并且 A/D 开始再次对当前选择的通道进行采样。如果采集时间已经被编程，那么将不会有任何指示显示采集时间何时结束，转换何时开始。

18.3 选择 A/D 转换时钟

每位的 A/D 转换时间被定义为 TAD。每完成一次 10 位 A/D 转换需要 11 个 TAD。可用软件选择 A/D 转换的时钟源。 TAD 可有以下 7 种选择：

- 2 TOSC
- 4 TOSC
- 8 TOSC
- 16 TOSC
- 32 TOSC
- 64 TOSC
- 内部 RC 振荡器

为了实现正确的 A/D 转换， A/D 转换时钟（TAD）必须尽可能得小，但它必须大于最小 TAD（大约为 2 μs，欲知更多信息请参见参数 130）。

表 18-1 显示了器件在不同的工作频率下和选择不同的 A/D 时钟源时得到的 TAD。

表 18-1： 不同器件工作频率下的 TAD

AD 时钟源 （TAD）		最高器件频率	
工作状态	ADCS2:ADCS0	PIC18F6X10/8X10	PIC18LF6X10/8X10[4]
2 TOSC	000	1.25 MHz	666 kHz
4 TOSC	100	2.50 MHz	1.33 MHz
8 TOSC	001	5.00 MHz	2.66 MHz
16 TOSC	101	10.0 MHz	5.33 MHz
32 TOSC	010	20.0 MHz	10.65 MHz
64 TOSC	110	40.0 MHz	21.33 MHz
RC[3]	x11	1.00 MHz[1]	1.00 MHz[2]

注 **1：** RC 源的典型 TAD 时间为 4 μs。
2： RC 源的典型 TAD 时间为 6 μs。
3： 当器件工作频率高于 1 MHz 时，整个转换过程必须在休眠模式下进行，否则 A/D 转换精度可能超出规范允许的范围。
4： 仅适用于低功耗器件 （PIC18LFXXXX）。

18.4 在功耗管理模式下的操作

在功耗管理模式中，自动采集时间和 A/D 转换时钟的选择一定程度上可由时钟源和频率决定。

如果希望器件处于功耗管理模式时进行 A/D 转换，ADCON2 中的 ACQT2:ACQT0 和 ADCS2:ADCS0 位就应该根据该模式下使用的时钟进行更新。在进入功耗管理模式之后，就可以开始 A/D 采集或转换。采集或转换开始以后，器件应继续使用相同的时钟源直到转换完成。如果需要的话，在转换期间也可以将器件置于相应的功耗管理空闲模式。

如果功耗管理模式下的时钟频率小于 1 MHz，就应该选择 A/D 模块的 RC 时钟源。

在休眠模式下工作需要选择 A/D 模块的 FRC 时钟。如果将 ACQT2:ACQT0 设置为 000 并启动 A/D 转换，转换将延迟一个指令周期以允许执行SLEEP指令并进入休眠模式。 OSCCON 寄存器中的 IDLEN 和 SCS 位必须在转换开始之前被清零。

18.5 配置模拟端口引脚

ADCON1、 TRISA 和 TRISF 寄存器均可用于配置 A/D 端口引脚。若希望端口引脚为模拟输入，则必须将相应的 TRIS 位置 1（输入）。如果将 TRIS 位清零（输出），则该引脚将输出数字电平 （VOH 或 VOL）。

A/D 转换与 CHS3:CHS0 位及 TRIS 位的状态无关。

注		
	1：	读取端口寄存器时，所有配置为模拟输入通道的引脚均读为 0 （低电平）。配置为数字输入的引脚将模拟输入电平精确转换为数字引脚电平。
	2：	定义为数字输入引脚上的模拟电平可能会导致数字输入缓冲器消耗的电流超出器件规范。

18.6 A/D 转换

图 18-4 显示了在 GO 位置 1 且 ACQT2:ACQT0 位被清零后 A/D 转换器的工作状态。转换在下一条指令执行之后开始，以允许器件在转换开始之前进入休眠模式。

图 18-5 显示了在 GO 位置 1， ACQT2:ACQT0 位被设置为 `010`，且在转换开始之前选择 4 TAD 采集时间后 A/D 转换器的工作状态。

在转换期间将 GO/$\overline{DONE}$ 位清零将中止当前的 A/D 转换。不会用尚未完成的 A/D 转换结果更新 A/D 结果寄存器对。这意味着 ADRESH:ADRESL 寄存器对仍将保持上一次转换的结果（和上一次写入 ADRESH:ADRESL 寄存器的值）。

在 A/D 转换完成或停止以后，需要等待 2 个 TAD 才能开始下一次采集。等待时间一到，将自动开始对所选通道进行采集。

> **注：** 不应在启动 A/D 模块的指令中将 GO/DONE 位置 1。

18.7 放电

放电过程用于对电容阵列的值进行初始化。在每次采样之前都会对此阵列放电。这一特性有助于优化单位增益放大器，因为每次需要重新为电容阵列充电，而不是根据以前测量的值进行充放电。

图 18-4： A/D 转换 TAD 周期 （ACQT<2:0> = `000`, TACQ = 0）

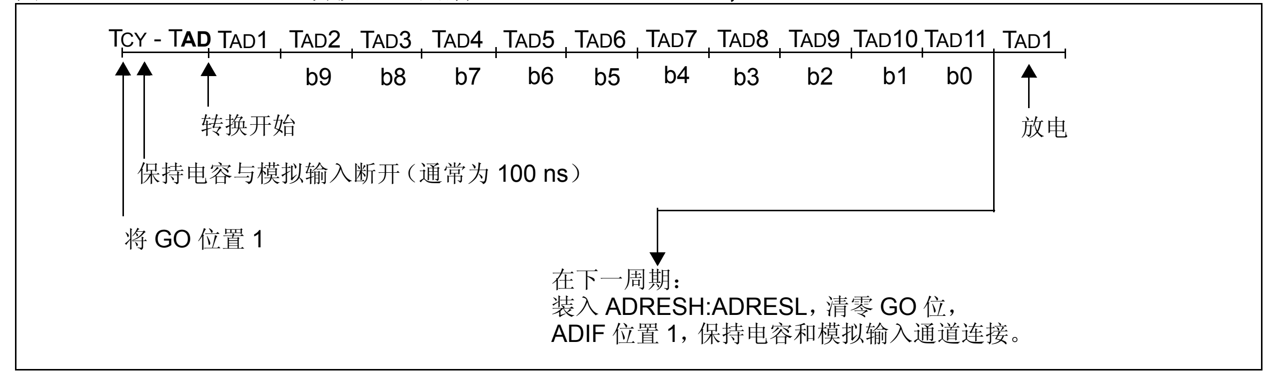
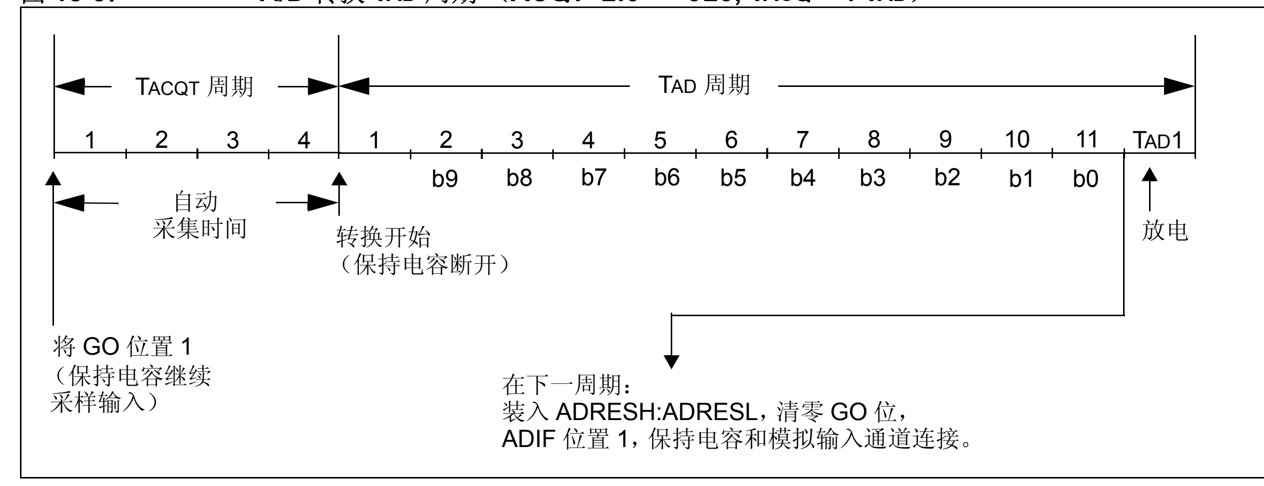


图 18-5： A/D 转换 TAD 周期 （ACQT<2:0> = `010`, TACQ = 4 TAD）



18.8 CCP2 触发器的使用

CCP2 模块的“特殊事件触发信号”可以启动 A/D 转换。这需要将 CCP2M3:CCP2M0 位（CCP2CON<3:0>）设置为 1011，且使能 A/D 模块（ADON 位置 1）。发生触发事件时，GO/DONE 位被置 1，启动 A/D 采样和转换并将 Timer1（或 Timer3）计数器复位为 0。复位 Timer1（或 Timer3）可自动重复 A/D 采集周期，最大限度地降低了软件开销（将 ADRESH/ADRESL 移到所需位置）。在“特殊事件触发信号”将 GO/DONE 位置 1（启动转换）之前，用户必须选择正确的模拟输入通道和最小采集时间，或选择合适的 TACQ 时间。

如果未使能 A/D 模块（ADON 清零），则“特殊事件触发信号”将被 A/D 模块忽略，但它仍会将 Timer1（或 Timer3）计数器复位。

表 18-2： 与 A/D 操作相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
PIR1	—	ADIF	RC1IF	TX1IF	SSPIF	CCP1IF	TMR2IF	TMR1IF	61
PIE1	—	ADIE	RC1IE	TX1IE	SSPIE	CCP1IE	TMR2IE	TMR1IE	61
IPR1	—	ADIP	RC1IP	TX1IP	SSPIP	CCP1IP	TMR2IP	TMR1IP	61
PIR2	OSCFIF	CMIF	—	—	BCLIF	HLVDIF	TMR3IF	CCP2IF	61
PIE2	OSCFIE	CMIE	—	—	BCLIE	HLVDIE	TMR3IE	CCP2IE	61
IPR2	OSCFIP	CMIP	—	—	BCLIP	HLVDIP	TMR3IP	CCP2IP	61
ADRESH	A/D 结果寄存器的高字节								61
ADRESL	A/D 结果寄存器的低字节								61
ADCON0	—	—	CHS3	CHS2	CHS1	CHS0	GO/DONE	ADON	61
ADCON1	—	—	VCFG1	VCFG0	PCFG3	PCFG2	PCFG1	PCFG0	61
ADCON2	ADFM	—	ACQT2	ACQT1	ACQT0	ADCS2	ADCS1	ADCS0	61
PORTA	RA7[(1)]	RA6[(1)]	RA5	RA4	RA3	RA2	RA1	RA0	62
TRISA	TRISA7[(1)]	TRISA6[(1)]	PORTA 数据方向寄存器						62
PORTF	读 PORTF 引脚，写 LATF 锁存器								62
TRISF	PORTF 数据方向寄存器								62
LATF	PORTF 输出数据锁存器								62

图注： — = 未用位，读为 0。 A/D 转换不使用阴影单元。

注 1： 可以根据所选择的振荡器模式将这些引脚配置为端口引脚。

 初稿

19.0 比较器模块

模拟比较器模块包含两个比较器，可以用多种方式对它们进行配置。该比较器的输入可以是与 RF3 到 RF6 引脚复用的模拟输入，也可以为片上参考电压（见**第 20.0 节 “比较器参考电压源模块”**）。数字输出（正常或翻转的）可从引脚电平获取也可通过控制寄存器读取。

CMCON 寄存器（寄存器 19-1）用于配置比较器的输入和输出。图 19-1 显示了各种比较器的配置。

寄存器 19-1： CMCON：比较器控制寄存器

R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-1	R/W-1	R/W-1
C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0
bit 7							bit 0

bit 7 **C2OUT**：比较器 2 输出位

当 C2INV = 0 时：
1 = C2 V_{IN+} > C2 V_{IN-}
0 = C2 V_{IN+} < C2 V_{IN-}

当 C2INV = 1 时：
1 = C2 V_{IN+} < C2 V_{IN-}
0 = C2 V_{IN+} > C2 V_{IN-}

bit 6 **C1OUT**：比较器 1 输出位

当 C1INV = 0 时：
1 = C1 V_{IN+} > C1 V_{IN-}
0 = C1 V_{IN+} < C1 V_{IN-}

当 C1INV = 1 时：
1 = C1 V_{IN+} < C1 V_{IN-}
0 = C1 V_{IN+} > C1 V_{IN-}

bit 5 **C2INV**：比较器 2 输出翻转位

1 = C2 输出翻转
0 = C2 输出不翻转

bit 4 **C1INV**：比较器 1 输出翻转位

1 = C1 输出翻转
0 = C1 输出不翻转

bit 3 **CIS**：比较器输入选择位

当 CM2:CM0 = 110 时：
1 = C1 V_{IN-} 连接到 RF5/AN10
C2 V_{IN-} 连接到 RF3/AN8
0 = C1 V_{IN-} 连接到 RF6/AN11
C2 V_{IN-} 连接到 RF4/AN9

bit 2-0 **CM2:CM0**：比较器模式位

图 19-1 给出了比较器的几种模式以及相应 CM2:CM0 位的设置。

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

19.1 比较器配置

图 19-1 给出了比较器的 8 种工作模式。CMCON 寄存器的 CM2:CM0 位用于选择模式。TRISF 寄存器控制比较器引脚的数据方向。如果改变比较器模式，由于存在特定的模式改变延迟（如第 **26.0** 节 “电气规范”所示）比较器的输出电平可能会在此延迟期间无效。

注：	改变比较器工作模式期间应禁止比较器中断，以免产生错误的中断。

图 19-1： 比较器 **I/O** 工作模式

A = 模拟输入，始终读为 0　　D = 数字输入　　CIS（CMCON<3>）= 比较器输入选择位
* 将 TRISF<2:1> 位置 1 会通过把引脚配置为输入引脚从而禁止比较器输出。

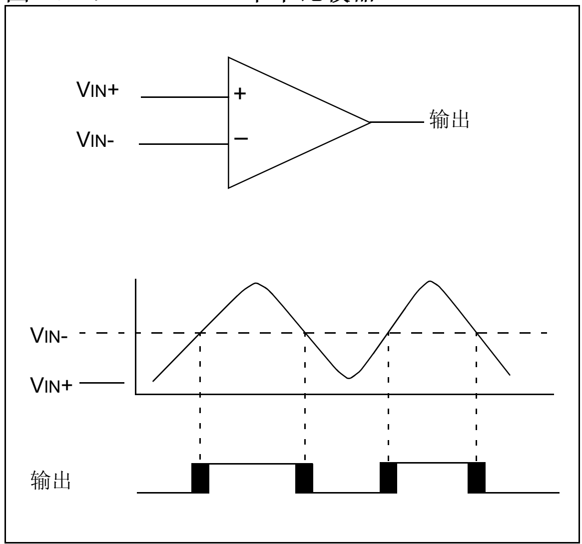
19.2 比较器的工作原理

图 19-2 显示了单个比较器，以及其模拟输入电平和数字输出之间的关系。如果 VIN+ 上的模拟输入电平值小于 VIN- 上的模拟输入值，那么比较器将输出数字低电平。当模拟输入端 VIN+ 上的模拟电压高于 VIN- 上的电压时，比较器输出数字高电平。图 19-2 中比较器输出的阴影部分表示因输入偏移和响应时间所造成的输出不确定区域。

19.3 比较器参考电压

根据不同的比较器工作模式，可选择使用外部或内部参考电压。将 VIN- 上的模拟信号和 VIN+ 上的信号做比较，并相应地调整比较器的数字输出 （图 19-2）。

图 19-2： 单个比较器



19.3.1 外部参考电压信号

当使用外部参考电压时，可将比较器模块中的两个比较器配置为使用同一个参考源或使用不同的参考源。然而，门限检测电路可能要求使用同一个参考源。参考信号幅值必须在 VSS 和 VDD 之间，并且可被施加到比较器的任一引脚上。

19.3.2 内部参考电压信号

比较器模块也可以选择使用内部参考电压模块产生的参考电压。在第 **20.0 节 "比较器参考电压源模块"** 中详细介绍了该模块。

只有在两个比较器复用四路输入的模式（CM2:CM0 = 110）中才可使用内部参考电压。在该模式下，内部参考电压被施加到两个比较器的 VIN+ 引脚上。

19.4 比较器的响应时间

响应时间是指从选定一个新的参考电压或输入源到比较器输出达到一个有效电平的最短时间。如果内部参考电压发生了改变，在使用比较器的输出时必须考虑到内部参考电压的最大延时。否则，应适用比较器的最大响应时间 （见第 **26.0 节 "电气规范"**）。

19.5 比较器输出

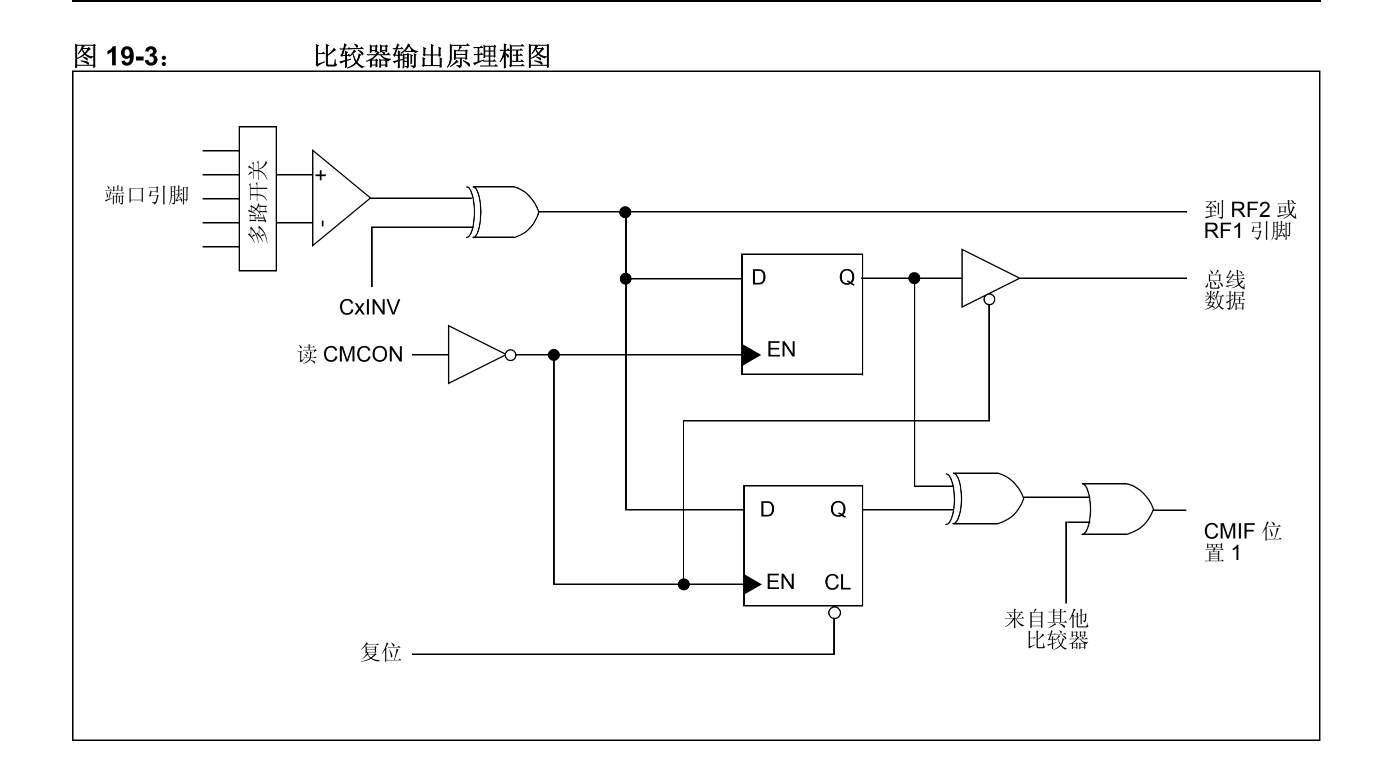
通过 CMCON 寄存器可读取比较器的输出。该寄存器是只读的。比较器的输出也可以直接输出到 I/O 引脚 RF2 和 RF1。当使能时， RF2 和 RF1 引脚输出路径上的多路开关会发生切换，并且每个引脚输出的信号与比较器输出信号是异步的。每个比较器输出的不确定区域的大小与电气规范中给出的输入偏移电压和响应时间有关。图 19-3 为比较器的输出原理框图。

在该模式下， TRISF 仍作为 RF2 和 RF1 引脚的输出使能 / 禁止位。

设置 C2INV 和 C1INV 位（CMCON<5:4>）可以改变比较器输出电压的极性。

注	
1：	读端口寄存器时，所有配置为模拟输入的引脚将读为 0。配置为数字输入的引脚将根据施密特触发器输入规范，对模拟输入电平进行相应的转换。
2：	定义为数字输入引脚上的模拟电平均可能会使输入缓冲器的电流消耗超过规定值。

图 19-3： 比较器输出原理框图



19.6 比较器中断

任何一个比较器的输出值一旦发生了变化，就会将该比较器的中断标志位置 1。为确定比较器输出是否发生了变化，需要用软件来保持输出位的状态信息（从 CMCON<7:6> 中读取数据）。CMIF 位（PIR2<6>）是比较器中断标志位，且必须通过清零复位。因为可以人为向CMCON寄存器写入1，所以也可以模拟中断产生。

必须将 CMIE 位（PIE2<6>）和 PEIE 位（INTCON<6>）置1以使能中断。此外，还必须将GIE位（INTCON<7>）置 1。只要这些位中的任何一位被清零，虽然当有中断条件产生时 CMIF 位仍会置 1，但中断是被禁止的。

注：	当执行读操作时（在 Q2 周期开始时），如果 CMCON 寄存器（C1OUT 或 C2OUT）的值发生了变化，那么 CMIF（PIR 寄存器）中断标志位可能不会被置 1。

用户可用以下方式在中断服务程序中清除该中断。

a） 对CMCON的任何读或写操作均将结束电平不匹配状态。

b) 将中断标志位 CMIF 清零。

引脚上电平的不匹配会不断地将 CMIF 标志位置 1。读 CMCON 寄存器将结束引脚上电平不匹配状态，并允许 CMIF 标志位清零。

19.7 在休眠模式下的比较器操作

当比较器处于运行模式而器件处于休眠模式时，比较器仍保持工作并可使用比较器中断（如果使能的话）。当使能中断时，中断会把器件从休眠模式唤醒。当比较器上电时会产生比规范中规定的掉电电流更高的休眠电流。每个比较器工作时都会消耗额外的电流，正如比较器规范中所示。若要把休眠状态下的功耗减少到最小，可在进入休眠模式前关闭比较器模块（CM2:CM0 = 111）。当器件从休眠模式被唤醒时，CMCON 寄存器的内容不受影响。

19.8 复位的影响

器件复位强制 CMCON 寄存器进入复位状态，从而使比较器模块进入比较器复位模式（CM2:CM0 = 000）。这确保所有的输入均为模拟输入。复位时引脚呈现模拟输入状态，器件电流降至最小。在复位期间，比较器模块断电。

 初稿

19.9 模拟输入连接注意事项

图 19-4 是一个简化的模拟输入电路。由于模拟引脚和数字输出端相连，因此在模拟引脚与 VDD 和 VSS 之间连有反向偏置的二极管，将其电压限制在 VSS 和 VDD 之间。一旦输入电压超出该范围 0.6V，其中一个二极管就会发生正向偏置从而使输入电压被钳位。模拟信号源的最大阻抗推荐值为 10 kΩ。任何连接到模拟输入引脚的外部元件 （如电容和齐纳二极管），要保证其漏电流极小。

图 19-4： 比较器模拟输入典型电路

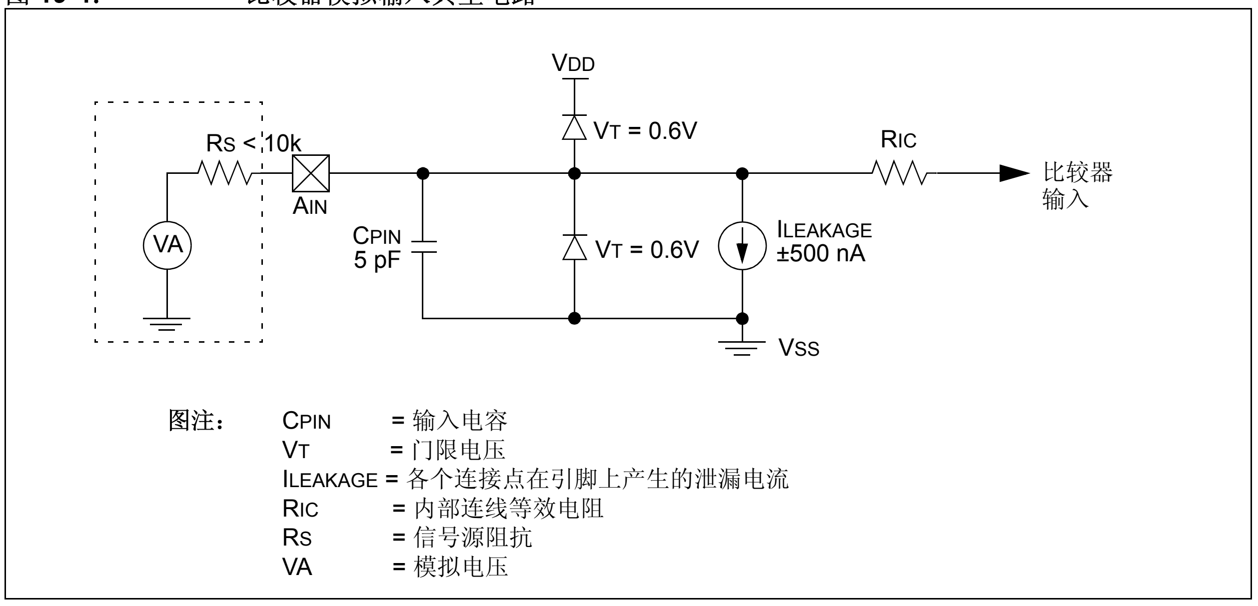


表 19-1： 与比较器模块相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
CMCON	C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0	61
CVRCON	CVREN	CVROE	CVRR	CVRSS	CVR3	CVR2	CVR1	CVR0	61
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
PIR2	OSCFIF	CMIF	—	—	BCLIF	HLVDIF	TMR3IF	CCP2IF	61
PIE2	OSCFIE	CMIE	—	—	BCLIE	HLVDIE	TMR3IE	CCP2IE	61
IPR2	OSCFIP	CMIP	—	—	BCLIP	HLVDIP	TMR3IP	CCP2IP	61
PORTF	读 PORTF 引脚，写 LATF 锁存器								62
LATF	LATF 数据输出寄存器								62
TRISF	PORTF 数据方向寄存器								62

图注： — = 未用位，读为 0。比较器模块不使用阴影单元。

注：

20.0 比较器参考电压源模块

比较器参考电压源模块是一个16阶的梯形电阻网络，可提供多个参考电压供选择。虽然它的主要目的是为模拟比较器提供参考电压，但也可将它用于其他场合。

图 20-1 显示了此模块的原理框图。梯形电阻经过分段可提供两种范围内的CVREF值，并且该网络还具有断电功能，可以在不使用参考电压的情况下节省功耗。器件的 VDD/VSS 或外部参考电压都可作为此模块的参考电源。

20.1 配置比较器参考电压源模块

参考电压源模块是通过 CVRCON 寄存器（寄存器 20-1）来控制的。比较器参考电压源模块提供两种范围的输出电压，每种范围都具有 16 个不同的电压值。CVRR 位（CVRCON<5>）选择输出电压的范围。这两种范围的主要区别在于其电压值之间的步长不同（其中一种范围可提供较高的分辨率），该步长由 VREF（CVR3:CVR0）位来决定。下面是计算比较器参考电压输出值的公式：

如果 CVRR = 1：
CVREF = ((CVR3:CVR0)/24) x CVRSRC

如果 CVRR = 0：
CVREF = (CVDD x 1/4) + (((CVR3:CVR0)/32) x CVRSRC)

参考电压源模块电源电压可以来自 VDD 和 VSS，也可以来自与 RA2 和 RA3 复用的外部 VREF+ 和 VREF-。电压源由 CVRSS 位（CVRCON<4>）选择。

在更改CVREF输出值时必须考虑比较器参考电压的稳定时间（见第 **26.0 节 "电气规范"** 中的表 26-3）。

寄存器 20-1： CVRCON：比较器参考电压源控制寄存器

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CVREN	CVROE(1)	CVRR	CVRSS	CVR3	CVR2	CVR1	CVR0
bit 7							bit 0

bit 7 **CVREN**：比较器参考电压源使能位
1 = CVREF 电路上电
0 = CVREF 电路断电

bit 6 **CVROE**：比较器 VREF 输出使能位 (1)
1 = CVREF 电平也从 RF5/AN10/CVREF/SEG23 引脚输出
0 = CVREF 电压与 RF5/AN10/CVREF/SEG23 引脚断开

bit 5 **CVRR**：比较器 VREF 范围选择位
1 = 0.00 CVRSRC 到 0.75 CVRSRC，步长为 CVRSRC/24
0 = 0.25 CVRSRC 到 0.75 CVRSRC，步长为 CVRSRC/32

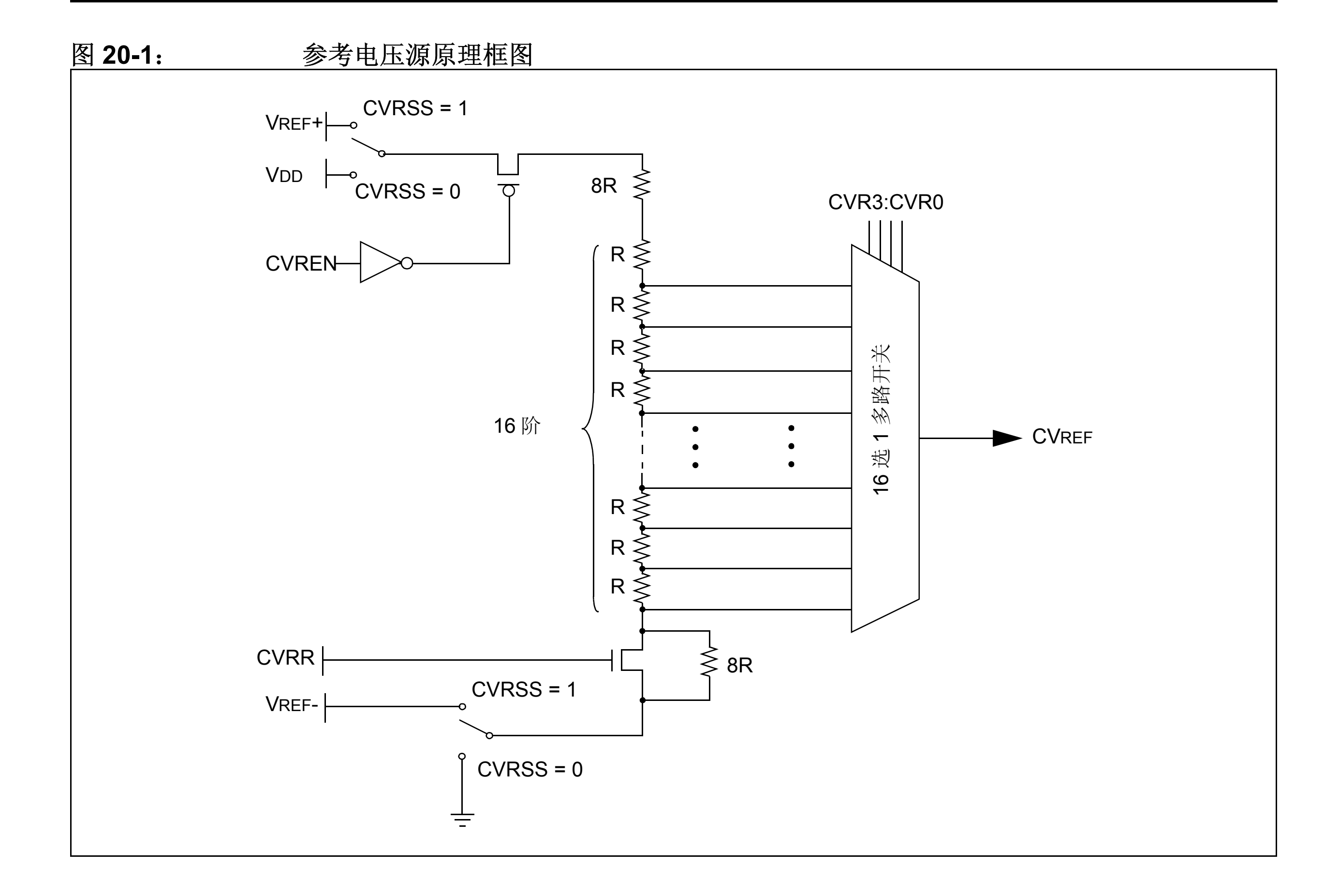
bit 4 **CVRSS**：比较器 VREF 源选择位
1 = 比较器参考电压源， CVRSRC = (VREF+) － (VREF-)
0 = 比较器参考电压源， CVRSRC = VDD － VSS

bit 3-0 **CVR3:CVR0**：比较器 VREF 值选择位（0 ≤ （CVR3:CVR0）≤ 15）
当 CVRR = 1 时：
CVREF = ((CVR3:CVR0)/24)•(CVRSRC)
当 CVRR = 0 时：
CVREF = (CVRSRC/4) + ((CVR3:CVR0)/32) •(CVRSRC)

注 1：当使能为输出时，CVROE 的优先级高于 TRISF<5> 位。必须通过将 TRISF<5> 置 1，把 RF5 配置为输入。

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
- n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

图 20-1： 参考电压源原理框图



20.2 参考电压精度 / 误差

由于模块结构的限制，并不能实现整个参考电压范围的满量程输出。梯形电阻网络顶部和底部的晶体管（图 20-1）使 CVREF 值不能达到参考电压源的上下限。参考电压是由电源电压分压而来的，因此 CVREF 输出随参考电压源的波动而变化。参考电压的绝对精度请参见第 **26.0** 节 **“电气规范”**。

20.3 休眠期间的操作

如果因中断或看门狗定时器超时将器件从休眠模式唤醒， CVRCON 寄存器的内容将不受影响。为了降低休眠模式下的电流消耗，应关闭参考电压模块。

20.4 复位的影响

器件复位时， CVREN 位 （CVRCON<7>）将被清零从而禁止参考电压模块。复位还将 CVROE 位（CVRCON<6>）清零，使参考电压与 RA2 引脚断开；同时通过将 CVRR 位（CVRCON<5>）清零选择高电压范围。CVR 值选择位也将清零。

20.5 连接注意事项

参考电压源模块的工作独立于比较器模块。如果 TRISF<5> 位和 CVROE 位都被置 1，那么参考电压发生器的输出与 RF5 引脚相连，此时如果该引脚上出现输入信号，将会增大电流消耗。使能 CVRSS 时，将 RF5 用作数字输出引脚也将增加电流消耗。

RF5 引脚可被直接用作 D/A 输出，但是其驱动能力有限。要提高电流驱动能力， VREF 输出端必须外接缓冲器。图 20-2 举例说明了这一缓冲技术。

 初稿

图 **20-2**： 参考电压输出缓冲示例

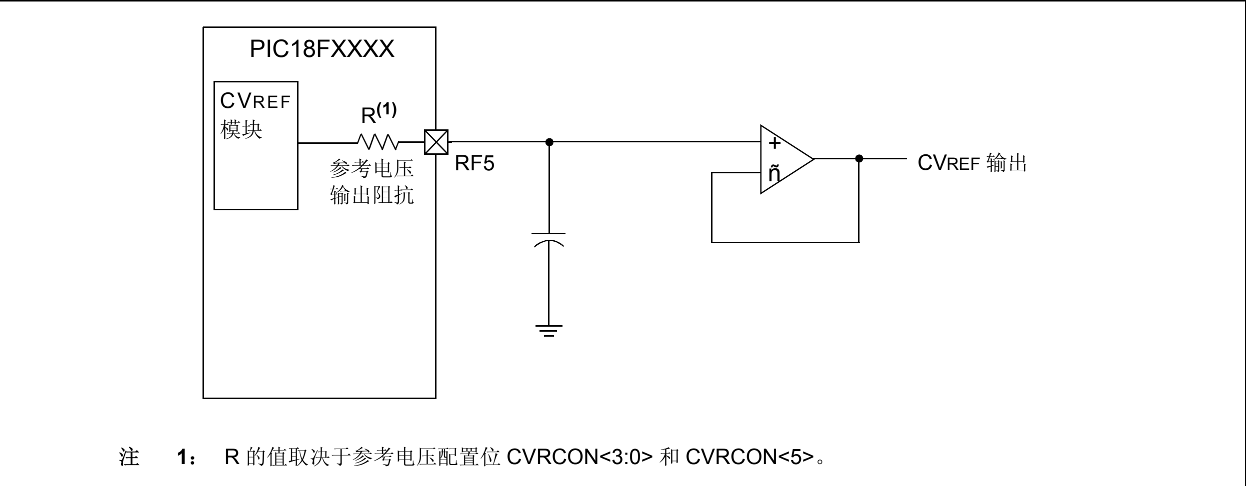


表 **20-1**： 与比较器参考电压源相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
CVRCON	CVREN	CVROE	CVRR	CVRSS	CVR3	CVR2	CVR1	CVR0	61
CMCON	C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0	61
TRISF	PORTF 数据方向寄存器								62

图注： 比较器参考电压源模块不使用阴影单元。

注：

21.0 高 / 低压检测 （HLVD）

PIC18F6390/6490/8390/8490 器件配有一个高 / 低压检测模块 （HLVD）。该模块是一个可编程的电路，它允许用户指定器件的电压跳变点和从该点起的电压变化方向。如果器件电压按照指定的方向相对于该跳变点发生了偏离，就会将中断标志位置 1。如果使能了中断，程序将跳转到中断向量地址处执行，由软件响应该中断。

高 / 低压检测控制寄存器（寄存器 21-1）控制 HLVD 模块的工作。用户通过软件写该寄存器将该电路 "关闭"，从而使器件的电流消耗降至最低。

图 21-1 给出了 HLVD 模块的原理框图。

寄存器 21-1： HLVDCON：高 / 低压检测控制寄存器

R/W-0	U-0	R-0	R/W-0	R/W-0	R/W-1	R/W-0	R/W-1
VDIRMAG	—	IRVST	HLVDEN	HLVDL3[1]	HLVDL2[1]	HLVDL1[1]	HLVDL0[1]
bit 7							bit 0

bit 7 **VDIRMAG**：电压大小方向选择位
1 = 当电压等于或超过跳变点 （HLVDL3:HLVDL0）时，事件发生。
0 = 当电压等于或跌落到跳变点 （HLVDL3:HLVDL0）以下时，事件发生。

bit 6 未用位：读为 0

bit 5 **IRVST**：内部参考电压稳定标志位
1 = 表示电压检测逻辑在检测到指定的电压范围时，产生中断标志
0 = 表示电压检测逻辑在检测到指定的电压范围时，不会产生中断标志，并且 HLVD 中断不被使能

bit 4 **HLVDEN**：高 / 低压检测源使能位
1 = 使能 HLVD
0 = 禁止 HLVD

bit 3-0 **HLVDL3:HLVDL0**：电压检测门限位 [1]
1111 = 使用外部模拟输入 （输入来自于 HLVDIN 引脚）
1110 = 4.41V-4.87V
1101 = 4.11V-4.55V
1100 = 3.92V-4.34V
1011 = 3.72V-4.12V
1010 = 3.53V-3.91V
1001 = 3.43V-3.79V
1000 = 3.24V-3.58V
0111 = 2.95V-3.26V
0110 = 2.75V-3.03V
0101 = 2.64V-2.92V
0100 = 2.43V-2.69V
0011 = 2.35V-2.59V
0010 = 2.16V-2.38V
0001 = 1.96V-2.16V
0000 = 保留

注 **1**：导致跳变点低于器件有效工作电压的 HLVDL3:HLVDL0 模式并未经过测试。

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

通过将 HLVDEN 位置 1 使能该模块。每次使能 HLVD 模块时，电路需要一定时间才能稳定下来。IRVST 位是一个只读位，用来表明电路是否稳定。仅当电路稳定且 IRVST 位置 1 后，该模块才能产生中断。

VDIRMAG 位决定该模块的整体工作状态。当 VDIRMAG 清零时，模块监视 VDD 看它是否跌落到预先确定的设置点以下。当该位置 1 时，模块监视 VDD 看它是否上升到设置点以上。

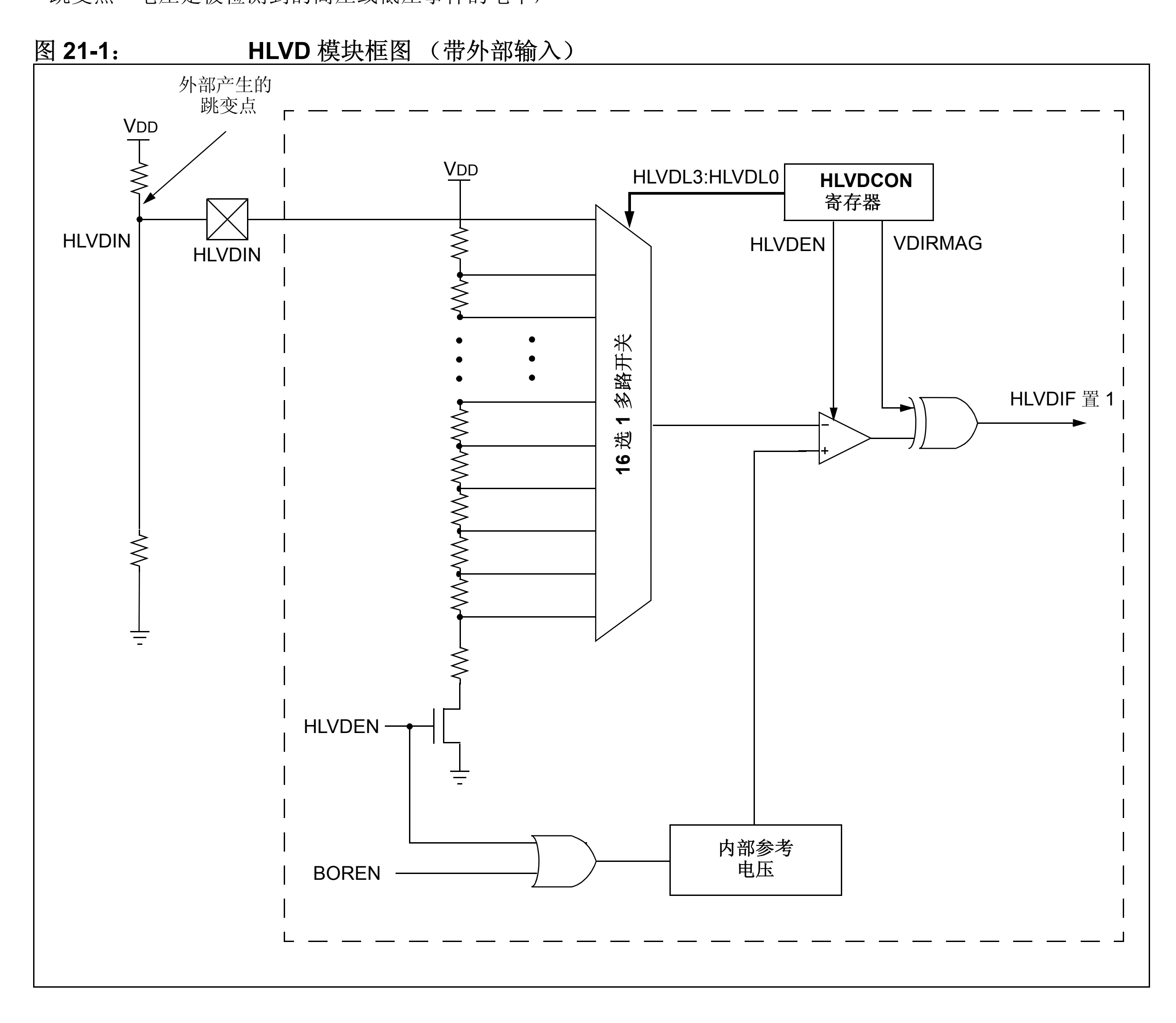
21.1 工作原理

当使能了 HLVD 模块后，比较器使用内部产生的参考电压作为设置点。将设置点的电压与跳变点电压作比较，其中电阻分压器中的每个节点均代表一个电压跳变点。“跳变点”电压是被检测到的高压或低压事件的电平，它取决于该模块的配置。当供电电压等于跳变点电压时，电阻阵列的分接电压输出值等于由参考电压源模块产生的内部参考电压。然后比较器通过将 HLVDIF 位置 1 产生一个中断信号。

可用软件编程指定跳变点电压为16个值中的任何一个。通过对 HLVDL3:HLVDL0 位 （HLVDCON<3:0>）进行编程可以选择跳变点。

HLVD 模块允许用户通过外部电源向模块提供跳变电压。当 HLVDL3:HLVDL0 位被设置为 “1111”时，使能此模式。在此状态下，比较器输入与外部输入引脚 HLVDIN 复用。因此用户可以灵活地配置高 / 低压检测中断，使之可以在有效工作范围内的任何电压点上产生。

图 21-1： HLVD 模块框图 （带外部输入）



21.2 设置 HLVD

要设置 HLVD 模块，需要遵循以下步骤：

1. 通过清零 HLVDEN 位（HLVDCON<4>）禁止该模块。
2. 将值写入 HLVDL3:HLVDL0，选择所需的 HLVD 跳变点。
3. 将 VDIRMAG 位设置为检测高压 （VDIRMAG = 1）或低压 （VDIRMAG = 0）。
4. 通过将 HLVDEN 位置 1，使能 HLVD 模块。
5. 清零 HLVD 中断标志位 （PIR2<2>），该位可能被上次中断置 1。
6. 如果需要中断，将 HLVDIE 和 GIE 位 （PIE<2> 和 INTCON<7>）置 1 使能 HLVD 中断。直到 IRVST 位也置 1 时才会发生中断。

21.3 电流消耗

使能了该模块就使能了 HLVD 比较器和分压器，并将消耗静态电流。电气规范中的参数 #D022B 给出了使能该模块时的电流总消耗。

HLVD 模块无需一直工作，工作与否取决于具体的应用。要降低电流消耗，只需要在检测电压时，短时间地使能HLVD电路，在检测完成之后马上禁止HLVD模块。

21.4 HLVD 启动时间

电气规范中的参数 #D420 规定了 HLVD 模块的内部参考电压，该参考电压也可供其他内部电路，例如可编程欠压复位电路使用。如果禁止了 HLVD 或其他使用参考电压的电路以降低器件的电流消耗，则参考电压电路将需要一段时间稳定下来以后才能可靠地检测低压或高压条件。HLVD 启动时间 TIRVST 与器件时钟速度无关。它由电气规范中的参数 36（表 26-10）规定。

直到 TIRVST 结束并且参考电压达到稳定后才会使能 HLVD 中断标志。由于这个原因，在此时间间隔期间，超出设置点的偏离可能不会被检测到。具体情形请参见图 21-2 或图 21-3。

图 21-2： 低压检测工作时序 （VDIRMAG = 0）

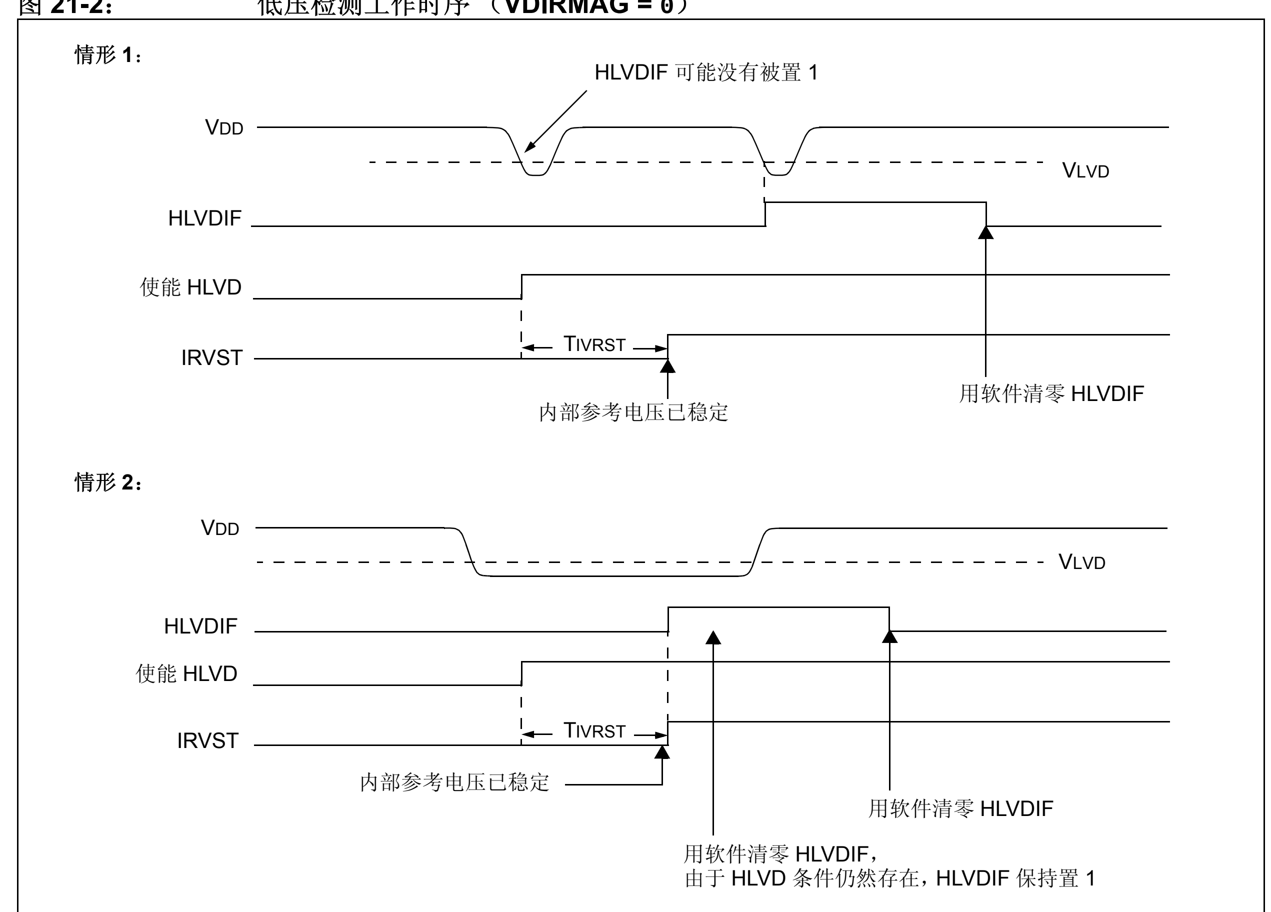


图 21-3： 高压检测工作时序 （VDIRMAG = 1）

情形 1：
HLVDIF 可能没有被置 1
VLVD
VDD
HLVDIF
使能 HLVD
IRVST
TIVRST
用软件清零 HLVDIF
内部参考电压已稳定

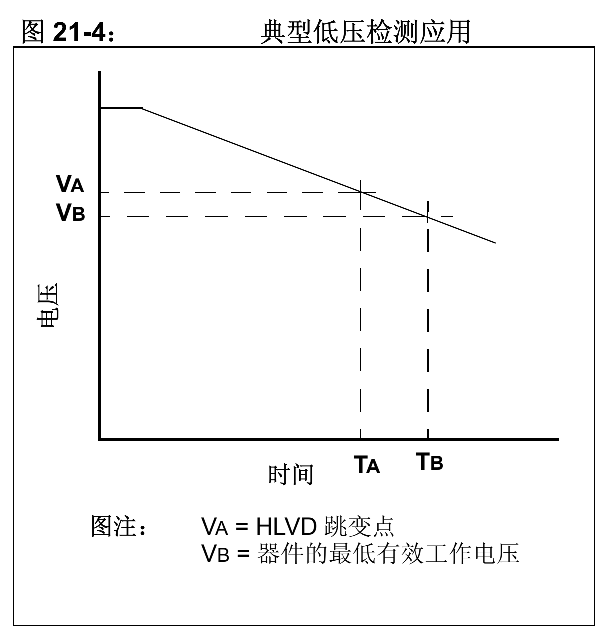
情形 2：
VLVD
VDD
HLVDIF
使能 HLVD
IRVST
TIVRST
内部参考电压已稳定
用软件清零 HLVDIF
用软件清零 HLVDIF，
由于 HLVD 条件仍然存在，HLVDIF 保持置 1

21.5 应用

在许多应用中，当电压低于或高于某个门限值时，系统希望可以检测到该事件。 例如，可以定期使能 HLVD 模块来检测是否连接 USB，这里假定器件的供电电压低于通用串行总线电压。若连接了 USB，将检测到 3.3V 到 5V 的高压 （USB 上的电压）；如果断开连接，情况正好相反。 此功能可以省去一些额外的元件和连接信号（输入引脚）。

对于一般的电池应用，图 21-4 给出了一个近似的电压曲线。器件电压会随时间逐渐下降。当器件电压达到电压 VA 时，HLVD 逻辑电路会在时间 TA 产生中断。该中断将导致执行中断服务子程序，从而使应用程序能在器件电压退出有效工作范围 （对应的时间为 TB）之前执行 “日常任务”，并安全关闭。 因此，HLVD 将会提供一个时间窗（表示为 TA 和 TB 的时间差）使应用程序能安全地退出。

图 21-4： 典型低压检测应用



21.6 休眠期间的操作

如果被使能时， HLVD 电路在休眠期间将继续工作。如果器件电压越过了跳变点， HLVDIF 位将会被置 1 并且器件将从休眠状态中被唤醒。如果已经使能了全局中断，程序将跳转到中断向量地址处继续执行。

21.7 复位的影响

器件复位强制所有寄存器进入复位状态。这会强制关闭 HLVD 模块。

表 21-1： 与高 / 低压检测模块相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在的
HLVDCON	VDIRMAG	—	IRVST	HLVDEN	HLVDL3	HLVDL2	HLVDL1	HLVDL0	60
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
PIR2	OSCFIF	CMIF	—	—	BCLIF	HLVDIF	TMR3IF	CCP2IF	61
PIE2	OSCFIE	CMIE	—	—	BCLIE	HLVDIE	TMR3IE	CCP2IE	61
IPR2	OSCFIP	CMIP	—	—	BCLIP	HLVDIP	TMR3IP	CCP2IP	61

图注： — = 未用，读作 0。 HLVD 模块不使用阴影单元。

注：

22.0 液晶显示 （LCD）驱动模块

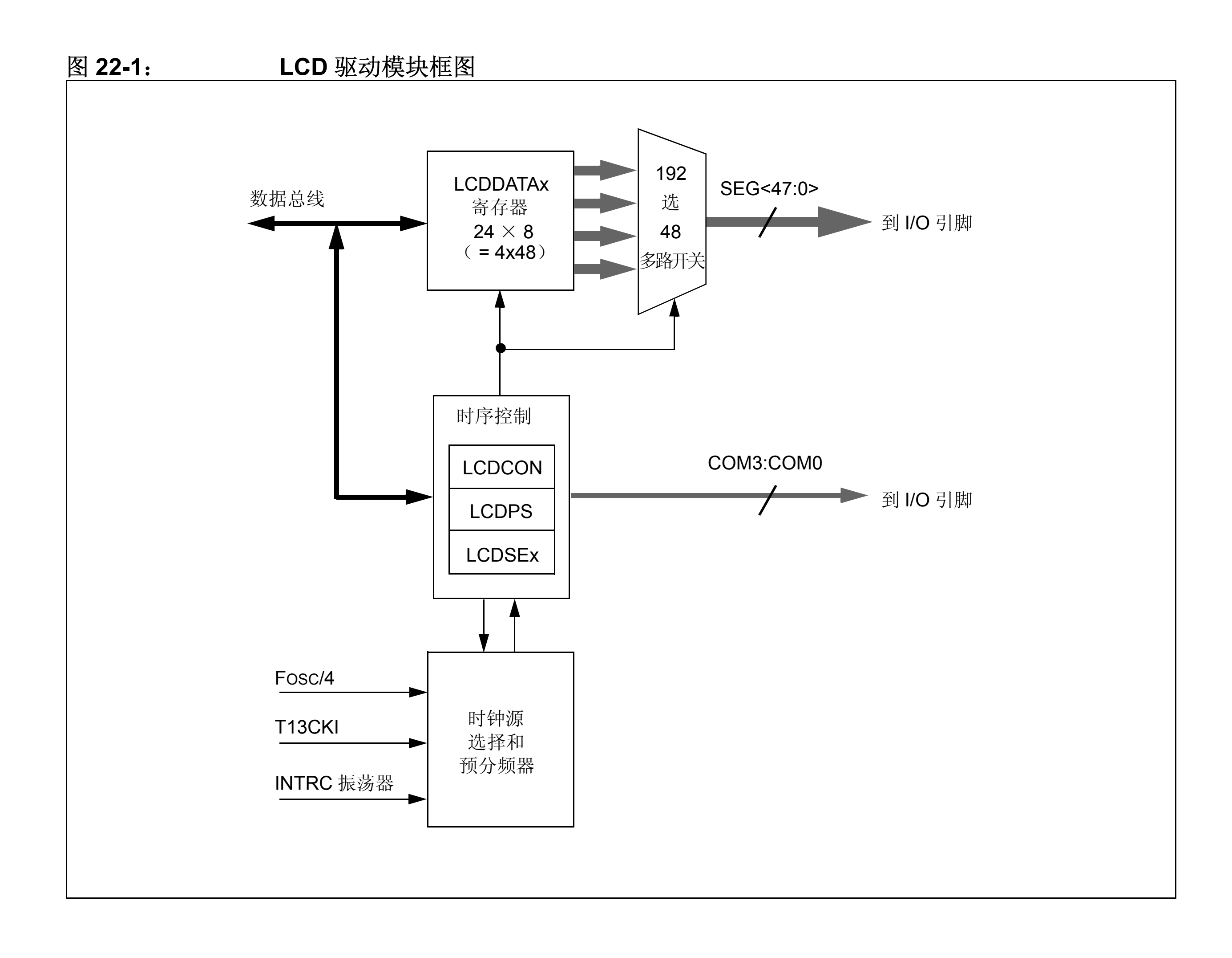
液晶显示（Liquid Crystal Dispaly， LCD）驱动模块产生时序控制驱动静态或复用的 LCD 面板。在 80 引脚器件中（PIC18F8390/8490），模块最多能驱动 4 公共端和 48 段的面板；在 64 引脚器件中（PIC18F6390/6490），模块最多能驱动 4 公共端和 32 段的面板。模块同时可以控制 LCD 像素数据。

LCD 驱动模块支持：

- LCD 面板的直接驱动
- 带有可选预分频比的 3 个 LCD 时钟源
- 多达 4 个公共端：
  - 静态
  - 1/2 复用
  - 1/3 复用
  - 1/4 复用
- 多达 48 （80 引脚器件中） /32 （64 引脚器件中）个段
- 静态、 1/2 或 1/3 LCD 偏置

图 22-1 所示为模块的简化框图。

图 22-1： LCD 驱动模块框图



22.1 LCD 寄存器

LCD 驱动模块包含 32 个寄存器：

- LCD 控制寄存器 （LCDCON）
- LCD 相位寄存器 （LCDPS）
- 6 个 LCD 段使能寄存器 （LCDSE5:LCDSE0）
- 24 个 LCD 数据寄存器 （LCDDATA23:LCDDATA0）

LCDCON 寄存器（如寄存器 22-1 所示）控制模块的整体操作。完成模块的配置之后，就可使用 LCDEN （LCDCON<7>）位使能或禁止 LCD 模块。通过清零 SLPEN （LCDCON<6>）位可使 LCD 面板工作在休眠模式下。

LCDPS 寄存器 （如寄存器 22-2 所示）用于配置 LCD 时钟源预分频比和波形 （A 型或 B 型）。关于这些功能的详细信息可参见**第 22.2 节 "LCD 时钟源选择"**、**第 22.3 节 "LCD 偏置类型"**以及**第 22.8 节 "LCD 波形的产生"**。

寄存器 22-1： LCDCON：LCD 控制寄存器

R/W-0	R/W-0	R/C-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
LCDEN	SLPEN	WERR	—	CS1	CS0	LMUX1	LMUX0
bit 7							bit 0

bit 7 **LCDEN**：LCD 驱动使能位
1 = 使能 LCD 驱动模块
0 = 禁止 LCD 驱动模块

bit 6 **SLPEN**：休眠模式下 LCD 驱动使能位
1 = 休眠模式下禁止 LCD 驱动模块
0 = 休眠模式下使能 LCD 驱动模块

bit 5 **WERR**：LCD 写失败错误位
1 = 当 LCDPS<WA> = 0 时写 LCDDATAx 寄存器 （必须用软件清零）
0 = 无 LCD 写错误

bit 4 **未用位**：读为 0

bit 3-2 **CS1:CS0**：时钟源选择位
00 = (Fosc/4)/8192
01 = T13CKI (Timer1)/32
1x = INTRC (31.25 kHz)/32

bit 1-0 **LMUX1:LMUX0**：公共端选择位

LMUX1:LMUX0	复用	最大像素数（PIC18F6X90）	最大像素数（PIC18F8X90）	偏置
00	静态 (COM0)	32	48	静态
01	1/2 (COM1:COM0)	64	96	1/2 或 1/3
10	1/3 (COM2:COM0)	96	144	1/2 或 1/3
11	1/4 (COM3:COM0)	128	192	1/3

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
C = 只可清零位	1 = 置 1	0 = 清零	x = 未知
-n = 上电复位时的值			

寄存器 22-2： LCDPS：LCD 相位寄存器

R/W-0	R/W-0	R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0
WFT	BIASMD	LCDA	WA	LP3	LP2	LP1	LP0
bit 7							bit 0

bit 7 **WFT**：波形选择位
1 = B 型波形 （在每一帧边缘改变相位）
0 = A 型波形 （在每一公共端类型内改变相位）

bit 6 **BIASMD**：偏置模式选择位
当 LMUX1:LMUX0 = 00 时：
0 = 静态偏置模式 （不要把该位置 1）
当 LMUX1:LMUX0 = 01 时：
1 = 1/2 偏置模式
0 = 1/3 偏置模式
当 LMUX1:LMUX0 = 10 时：
1 = 1/2 偏置模式
0 = 1/3 偏置模式
当 LMUX1:LMUX0 = 11 时 ：
0 = 1/3 偏置模式 （不要把该位置 1）

bit 5 **LCDA**：LCD 工作状态位
1 = 使能 LCD 驱动模块
0 = 禁止 LCD 驱动模块

bit 4 **WA**：LCD 写允许状态位
1 = 允许写入 LCDDATAx 寄存器
0 = 禁止写入 LCDDATAx 寄存器

bit 3-0 **LP3:LP0**：LCD 预分频比选择位
1111 = 1:16
1110 = 1:15
1101 = 1:14
1100 = 1:13
1011 = 1:12
1010 = 1:11
1001 = 1:10
1000 = 1:9
0111 = 1:8
0110 = 1:7
0101 = 1:6
0100 = 1:5
0011 = 1:4
0010 = 1:3
0001 = 1:2
0000 = 1:1

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

LCDSE5:LCDSE0 寄存器用于配置端口引脚的功能。通过置位特定段的段使能位，把该引脚配置为LCD驱动引脚。表 22-1列出了6个LCD段使能寄存器。寄存器 22-3 所示为 LCDSE 寄存器原型。

表 22-1： LCDSE 寄存器和相关段

寄存器	段
LCDSE0	7:0
LCDSE1	15:8
LCDSE2	23:16
LCDSE3	31:24
LCDSE4	39:32
LCDSE5	47:40

.

注：	PIC18F6X90 器件不包含 LCDSE5:LCDSE4 寄存器。

一旦初始化 LCD 面板模块， LCDDATA23:LCDDATA0 寄存器中的各位就会被清零或置位，分别代表透明和不透明像素。特定的段和公共端信号使用了一组特定的 LCDDATA 寄存器。寄存器中每一位均表示特定段同特定公共端间的唯一组合。 LCDDATA 位命名约定为 "SxxCy"，其中 "xx" 表示段号，"y" 表示公共端号。表 22-2 概括了两者间的关系，而寄存器 22-4 所示为 LCDDATA 寄存器原型。

注：	向 PIC18F6X90 器件的 LCDDATA4，LCDDATA5，LCDDATA10，LCDDATA11，LCDDATA16，LCDDATA17， LCDDATA22 和 LCDDATA23 寄存器写入数据不会影响任何像素的状态，可把这些寄存器用作通用寄存器。

寄存器 22-3： LCDSEx：LCD 段使能寄存器

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
SE(n + 7)	SE(n +6)	SE(n +5)	SE(n +4)	SE(n +3)	SE(n +2)	SE(n +1)	SE(n)
bit 7							bit 0

bit 7-0 **SEG(n + 7):SEG(n):** 段使能位

对于 LCDSE0: n = 0
对于 LCDSE1: n = 8
对于 LCDSE2: n = 16
对于 LCDSE3: n = 24
对于 LCDSE4: n = 32
对于 LCDSE5: n = 40
1 = 使能引脚的段功能，禁止其数字 I/O 功能
0 = 使能引脚的 I/O 功能

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

表 **22-2：** **LCDDATA** 寄存器位和段与公共端组合的对应关系

段	公共端			
	0	**1**	**2**	**3**
0 到 7	LCDDATA0	LCDDATA6	LCDDATA12	LCDDATA18
	S00C0:S07C0	S00C1:S07C1	S00C2:S07C2	S00C3:S07C3
8 到 15	LCDDATA1	LCDDATA7	LCDDATA13	LCDDATA19
	S08C0:S15C0	S08C1:S15C1	S08C2:S15C2	S08C0:S15C3
16 到 23	LCDDATA2	LCDDATA8	LCDDATA14	LCDDATA20
	S16C0:S23C0	S16C1:S23C1	S16C2:S23C2	S16C3:S23C3
24 到 31	LCDDATA3	LCDDATA9	LCDDATA15	LCDDATA21
	S24C0:S31C0	S24C1:S31C1	S24C2:S31C2	S24C3:S31C3
32 到 39	LCDDATA4(1)	LCDDATA10(1)	LCDDATA16(1)	LCDDATA22(1)
	S32C0:S39C0	S32C1:S39C1	S32C2:S39C2	S32C3:S39C3
40 到 47	LCDDATA5(1)	LCDDATA11(1)	LCDDATA17(1)	LCDDATA23(1)
	S40C0:S47C0	S40C1:S47C1	S40C2:S47C2	S40C3:S47C3

注 **1：** 64 引脚器件包含这些寄存器，但它们并不用作 LCD 数据寄存器，而是用作通用数据存储器。

寄存器 **22-4：** **LCDDATAx：LCD** 数据寄存器

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
S(n + 7)Cy	S(n +6)Cy	S(n +5)Cy	S(n +4)Cy	S(n +3)Cy	S(n +2)Cy	S(n +1)Cy	S(n)Cy
bit 7							bit 0

bit 7-0 **S(n + 7)Cy:S(n)Cy：**像素点亮位

对于 LCDDATA0:LCDDATA5：n = (8x)， y = 0

对于 LCDDATA6:LCDDATA11：n = (8(x-6))， y = 1

对于 LCDDATA12:LCDDATA17：n = (8(x-12))， y = 2

对于 LCDDATA18:LCDDATA23：n = (8(x-18))， y = 3

1 = 点亮像素 （不透明）

0 = 不点亮像素 （透明）

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

22.2 LCD 时钟源选择

LCD 驱动模块具有 3 种可用时钟源：

- (Fosc/4)/8192
- T13CKI-Clock/32
- INTRC/32

第一个时钟源是系统时钟的 8192 分频时钟信号((Fosc/4)/8192)。当系统时钟为 8 MHz 时，该分频比提供约 1kHz 的输出。该分频器不可编程。而 LCD 预分频器位 LCDPS<3:0> 是用来设置 LCD 帧时钟速率的。

第二个时钟源是 Timer1 振荡器的 32 分频时钟信号。当 Timer1 振荡器使用 32.768kHz 晶振时它同样提供约 1kHz 的输出。要把 Timer1 振荡器用作时钟源，需将 T1OSCEN （T1CON<3>）位置 1。

第三个时钟源是 31.25kHz 内部 RC 振荡器的 32 分频时钟信号，提供约 1kHz 输出。

当处理器休眠时，第二个和第三个时钟源用于维持 LCD 运行。

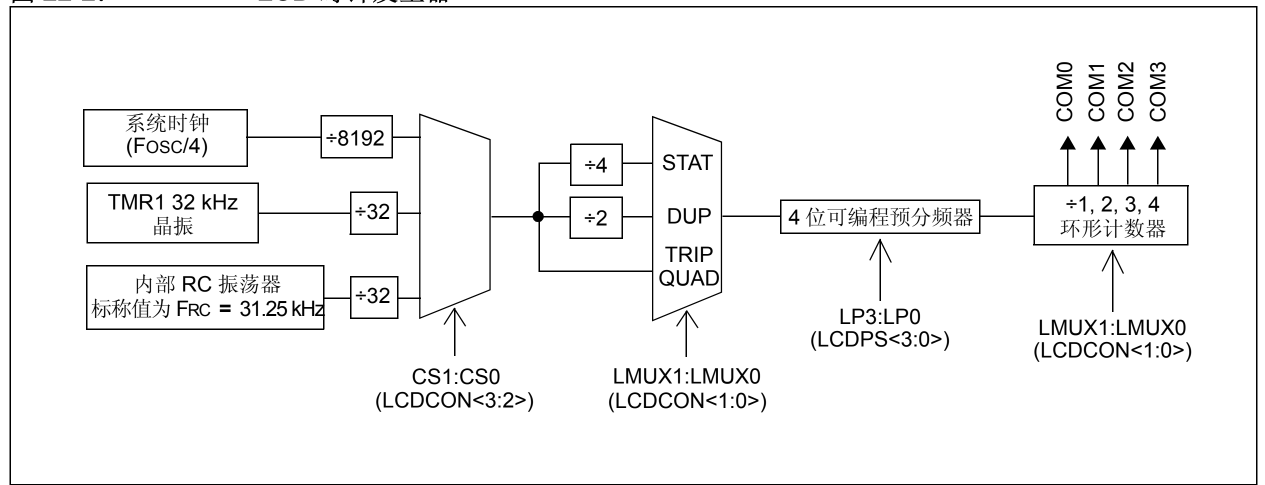
使用 CS1:CS0（LCDCON<3:2>）位，可选择这些时钟源中的一个。

22.2.1 LCD 预分频器

有一个 16 位计数器用作 LCD 时钟的预分频器，该预分频器不可直接读写。其值由 LP3:LP0 位（LCDPS<3:0>）设置，这些位决定预分频器的分配和预分频比值。

预分频比值从 1:1 到 1:16，以 2 的整数次幂为台阶递增。

图 22-2： LCD 时钟发生器



22.3 LCD 偏置类型

LCD 驱动模块可以被配置为 3 种偏置类型：

- 静态偏置 （2 个电压：AVSS 和 AVDD）
- 1/2 偏置 （3 个电压：AVSS、 1/2 AVDD 和 AVDD）
- 1/3 偏置 （4 个电压：AVSS、 1/3 AVDD、 2/3 AVDD 和 AVDD）

模块使用外部梯形电阻产生 LCD 偏置电压。

该外部梯形电阻应与偏置引脚 1、偏置引脚 2、偏置引脚 3 和 VSS 相连。还应把偏置引脚 3 连到 VDD。

图 22-3 给出了梯形电阻和偏置引脚的正确连法。

22.4 LCD 复用类型

LCD 驱动模块可以被配置为 4 种复用类型：

- 静态 （只使用 COM0）
- 1/2 复用 （使用 COM0 和 COM1）
- 1/3 复用 （使用 COM0、 COM1 和 COM2）
- 1/4 复用（使用 COM0、COM1、COM2 和 COM3）

LMUX1:LMUX0 的设置决定了 PORTE<6:4> 的功能（见表 22-3 了解详细信息）。

如果引脚为数字 I/O，对应的 TRIS 位控制数据方向。如果引脚为 COM 驱动，那么该引脚的 TRIS 设置将不起作用。

> **注：** 在上电复位时，LMUX1:LMUX0 位为 00。

表 22-3： PORTE<6:4> 功能

LMUX1:LMUX0	PORTE<6>	PORTE<5>	PORTE<4>
00	数字 I/O	数字 I/O	数字 I/O
01	数字 I/O	数字 I/O	COM1 驱动器
10	数字 I/O	COM2 驱动器	COM1 驱动器
11	COM3 驱动器	COM2 驱动器	COM1 驱动器

22.5 段使能

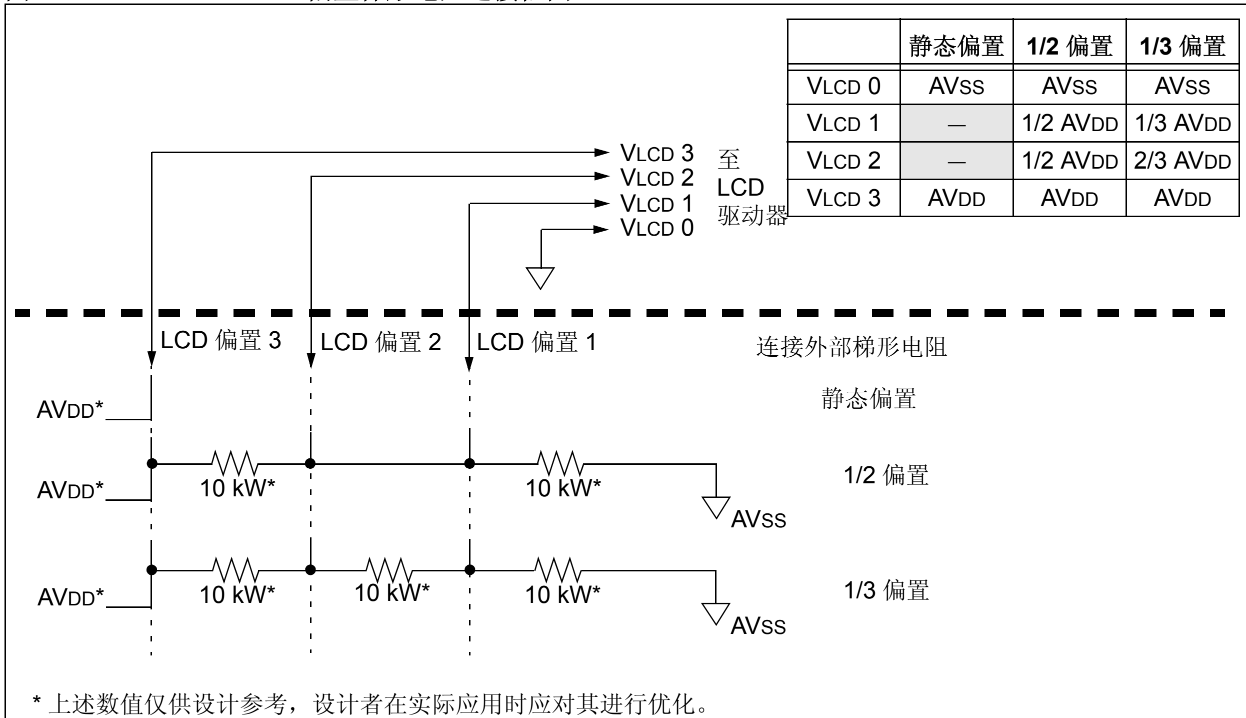
LCDSEx 寄存器用于选择每个段引脚的引脚功能。每个引脚可被配置为 LCD 段驱动器或数字引脚。要把引脚配置为段引脚， LCDSEx 寄存器中对应的位必须置 1。

如果引脚为数字 I/O，对应的 TRIS 位控制数据方向。LCDSEx 寄存器中的任何一位被置位，都会屏蔽 TRIS 中相应位的设置。

> **注：** 在上电复位时，这些引脚被配置为数字 I/O。



图 22-3： LCD 偏置梯形电阻连接框图



22.6 像素控制

LCDDATAx 寄存器中的位用于定义像素状态。每一位只定义一个像素。

表 22-2 所示为 LCDDATAx 寄存器中每一位同各个公共端和段信号间的相互关系。

没有用于显示的 LCD 像素地址可用作通用 RAM。

22.7 LCD 帧频率

COM 和 SEG 输出改变的速率称为 LCD 帧频率

表 22-4： 帧频率计算公式

复用	帧频率
静态	时钟源 / (4 x 1 x (LP3:LP0 + 1))
1/2	时钟源 / (2 x 2 x (LP3:LP0 + 1))
1/3	时钟源 / (1 x 3 x (LP3:LP0 + 1))
1/4	时钟源 / (1 x 4 x (LP3:LP0 + 1))

注： 时钟源为 （FOSC/4） /8192、Timer1 Osc/32 或 INTRC/32。

表 22-5： 近似帧频率 （单位 **HZ**）使用 **FOSC@32MHZ**，**TIMER1@32.768KHZ** 或者 **INTRC** 振荡器

LP3:LP0	静态	1/2	1/3	1/4
1	125	125	167	125
2	83	83	111	83
3	62	62	83	62
4	50	50	67	50
5	42	42	56	42
6	36	36	48	36
7	31	31	42	31

22.8 LCD 波形的产生

LCD 波形产生基于如下理论：不透明像素上的净 AC 电压应该是最大值而透明像素上的净 AC 电压应该为最小值。任何像素上的 DC 电压应该为零。

COM 信号表示每个公共端的时间片，而 SEG 中包含像素数据。

像素信号 （COM-SEG）中不包含直流分量，并且只可取两个 rms 值中的一个。高 rms 值会产生不透明像素而低 rms 值产生透明像素。

随着公共端数量的增加，两个 rms 值间的判别比逐渐减小。判别比表示显示器可具有的最大对比度。

可以用两种波形驱动 LCD：A 型和 B 型。在 A 型波形中，相位在每个公共端类型中改变，然而在 B 型波形中，相位在每个帧边界上改变。这样， A 型波形在单帧上维持 0 VDC 而 B 型波形则需要两个帧。

> 注 1： 如果要通过使能LCD休眠模式（LCDCON<SLPEN> =1）进入休眠状态，则必须格外小心，因为只有当所有像素上的 VDC 为 0 时才可执行休眠。
>
> 2： 当 LCD 时钟源是（FOSC/4）/8192 时，不管LCDCON<SLPEN> 设置如何，只要执行休眠指令，LCD 就会进入休眠状态。因而在执行休眠指令前，应注意查看所有像素上的 VDC 是否为 0。

图 22-4 到图 22-14 所示为 A 型和 B 型波形在静态、1/2 复用、 1/3 复用和 1/4 复用驱动时的波形图。

图 22-4： 在静态驱动时的 A/B 型波形

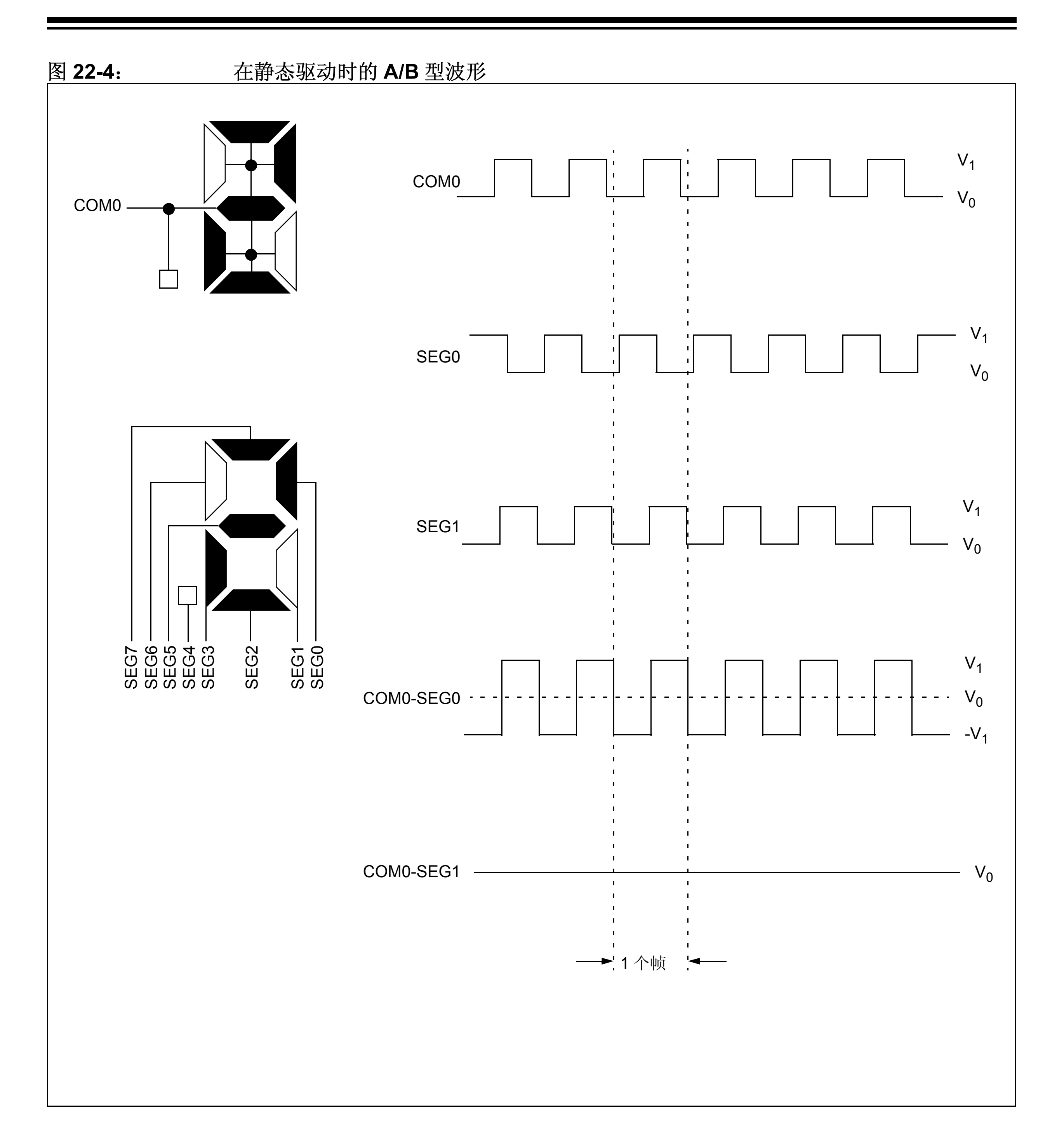
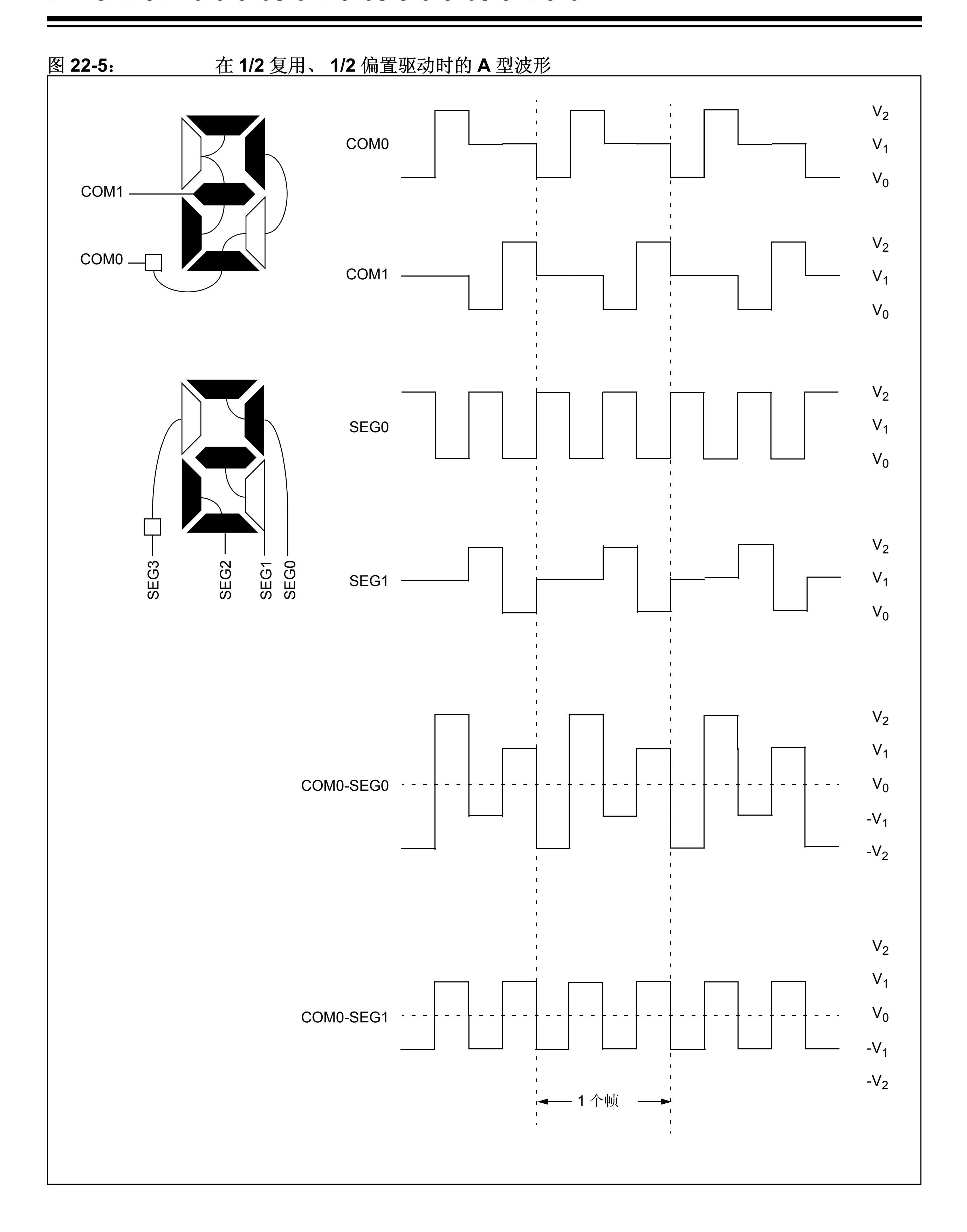


图 22-5： 在 1/2 复用、 1/2 偏置驱动时的 A 型波形



 初稿

图 22-6： 在 1/2 复用、 1/2 偏置驱动时的 B 型波形

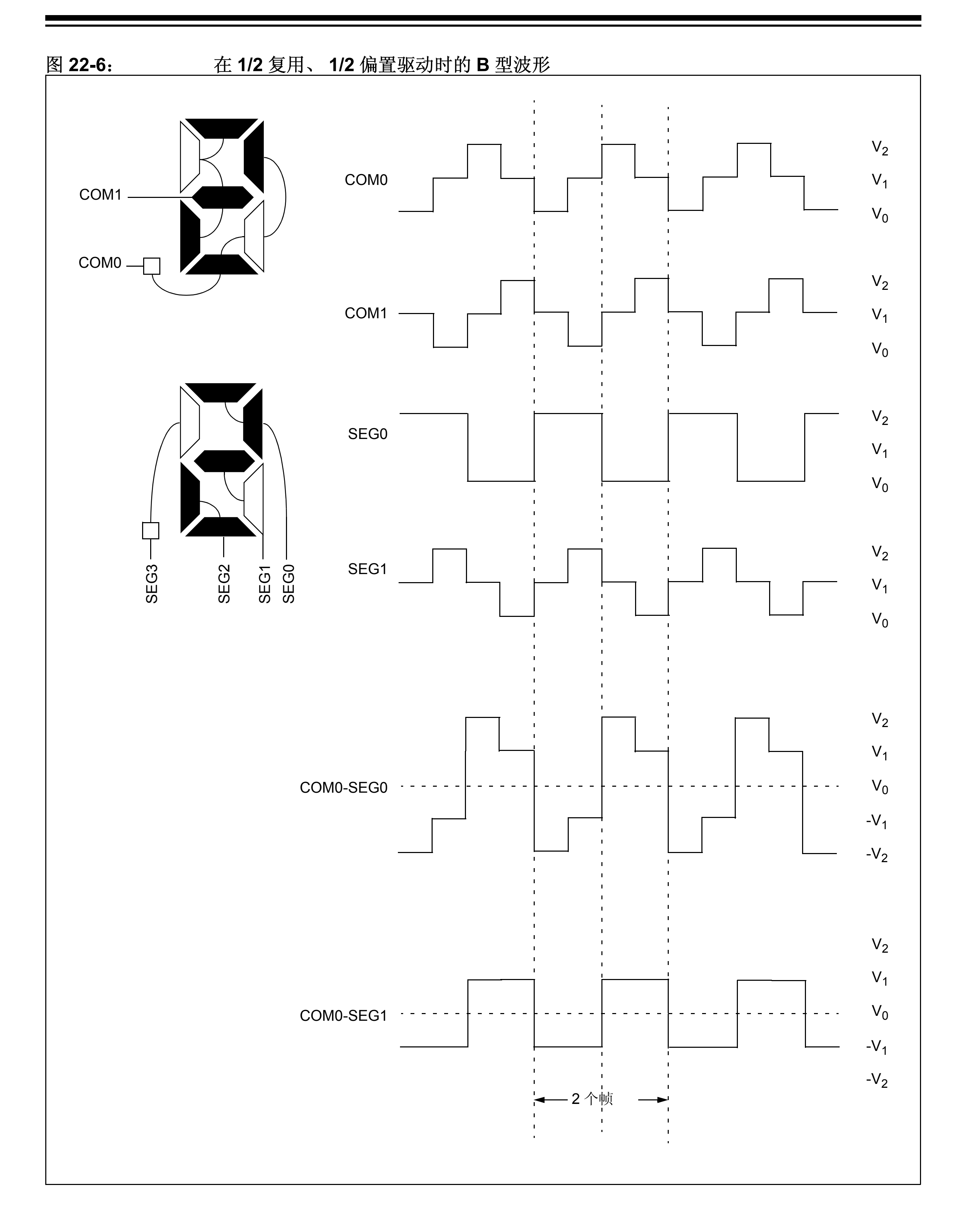
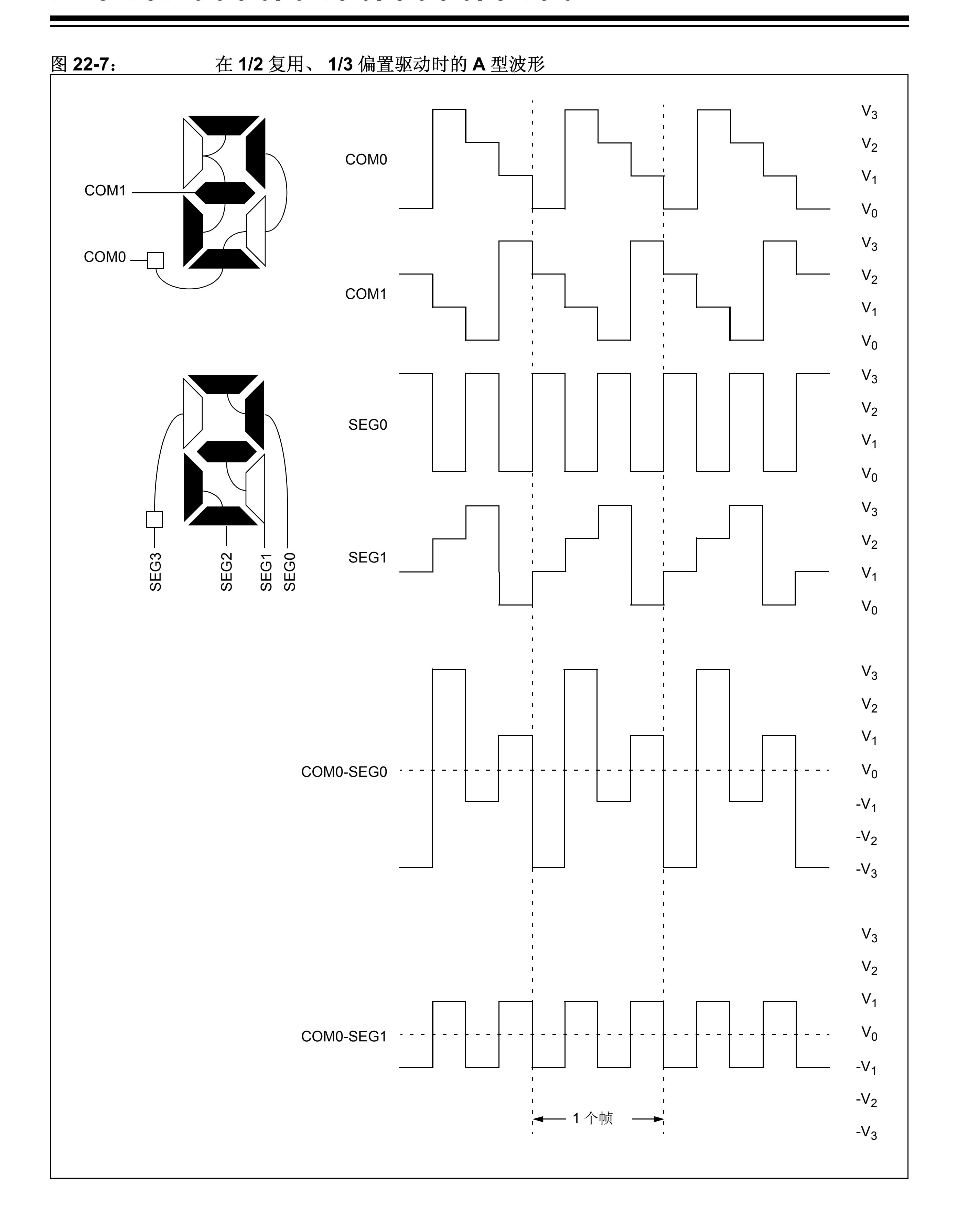


图 22-7： 在 1/2 复用、 1/3 偏置驱动时的 A 型波形



 初稿

图 22-8： 在 1/2 复用、 1/3 偏置驱动时的 B 型波形

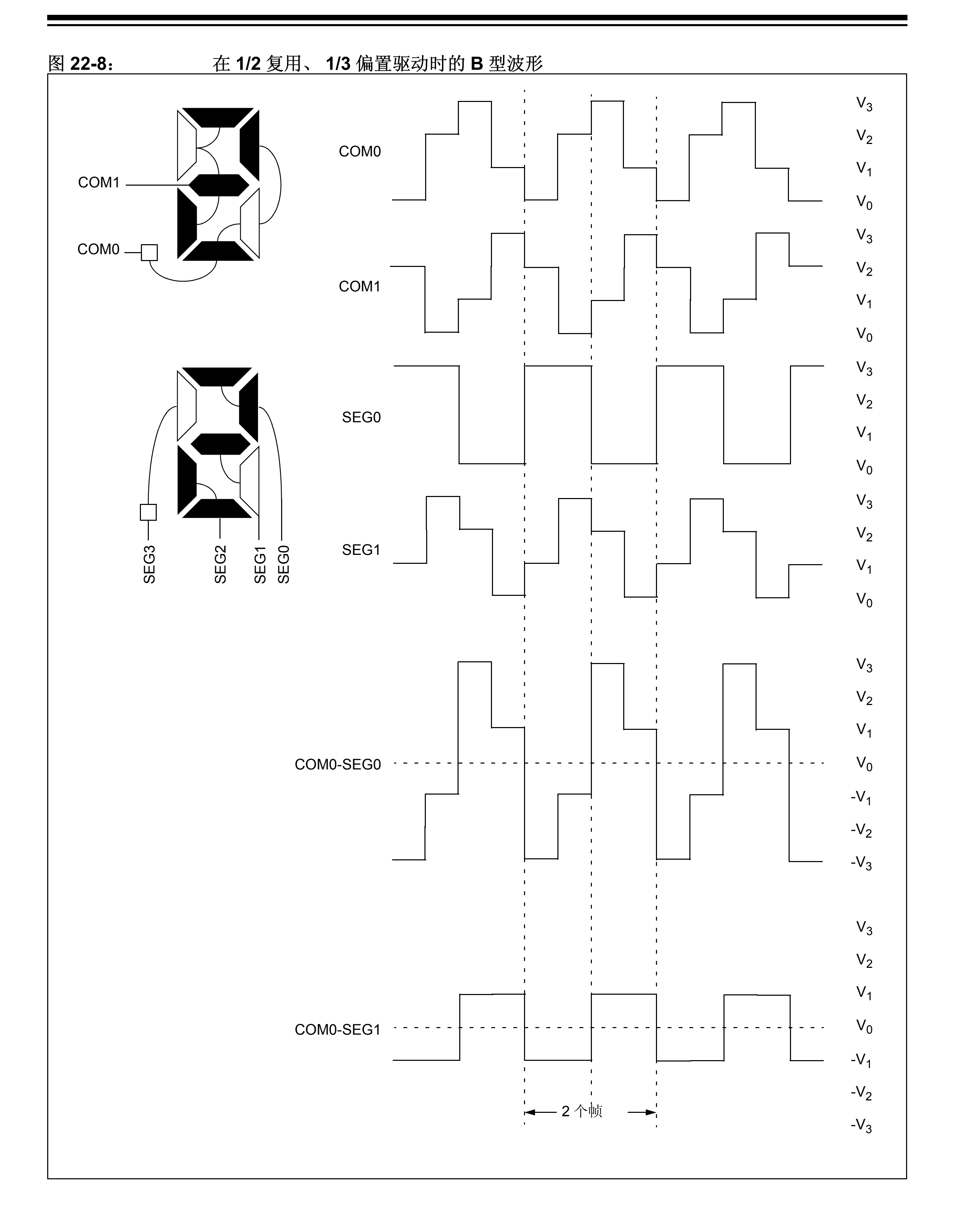
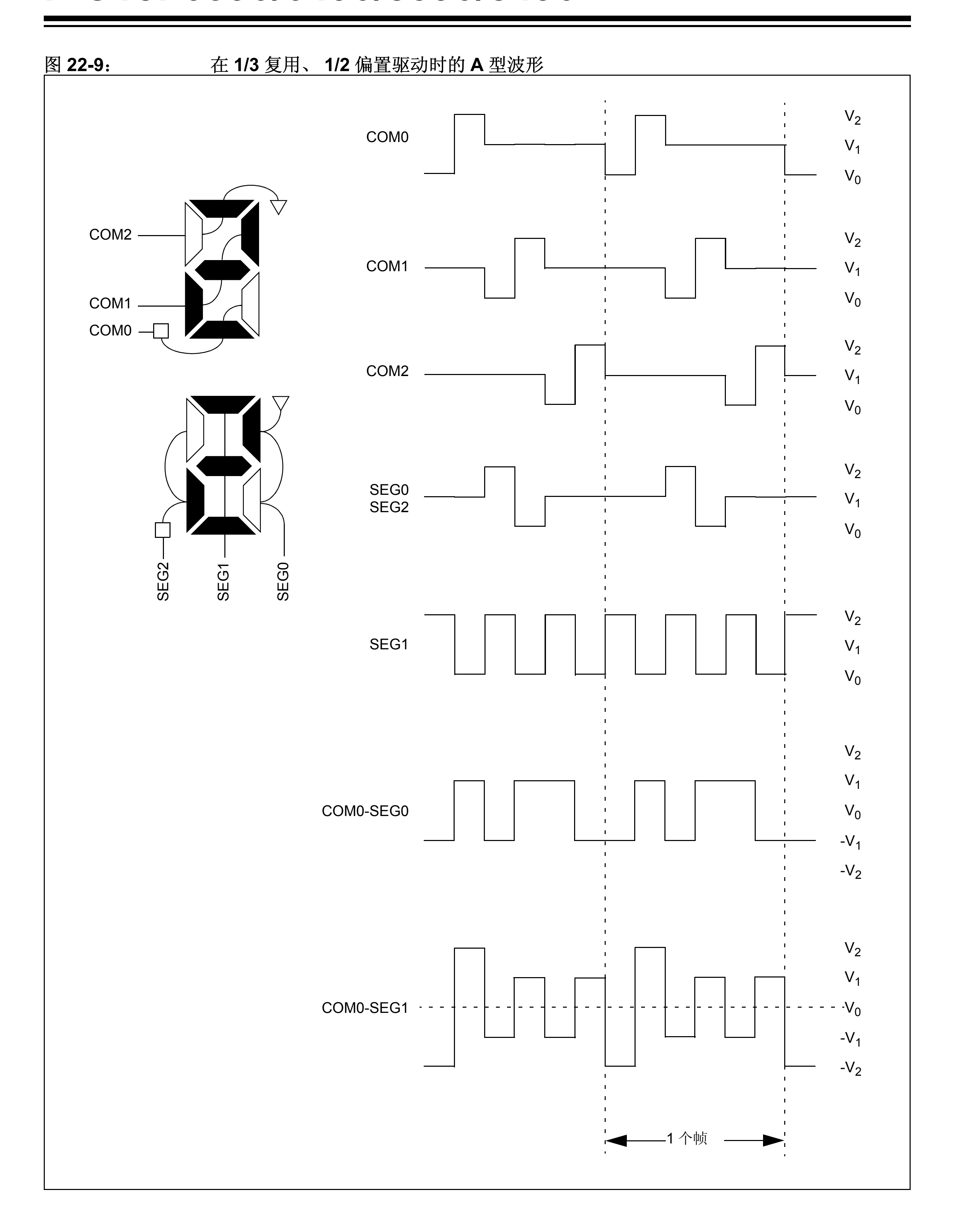


图 22-9： 在 1/3 复用、 1/2 偏置驱动时的 A 型波形



 初稿

图 **22-10**： 在 **1/3** 复用、 **1/2** 偏置驱动时的 **B** 型波形

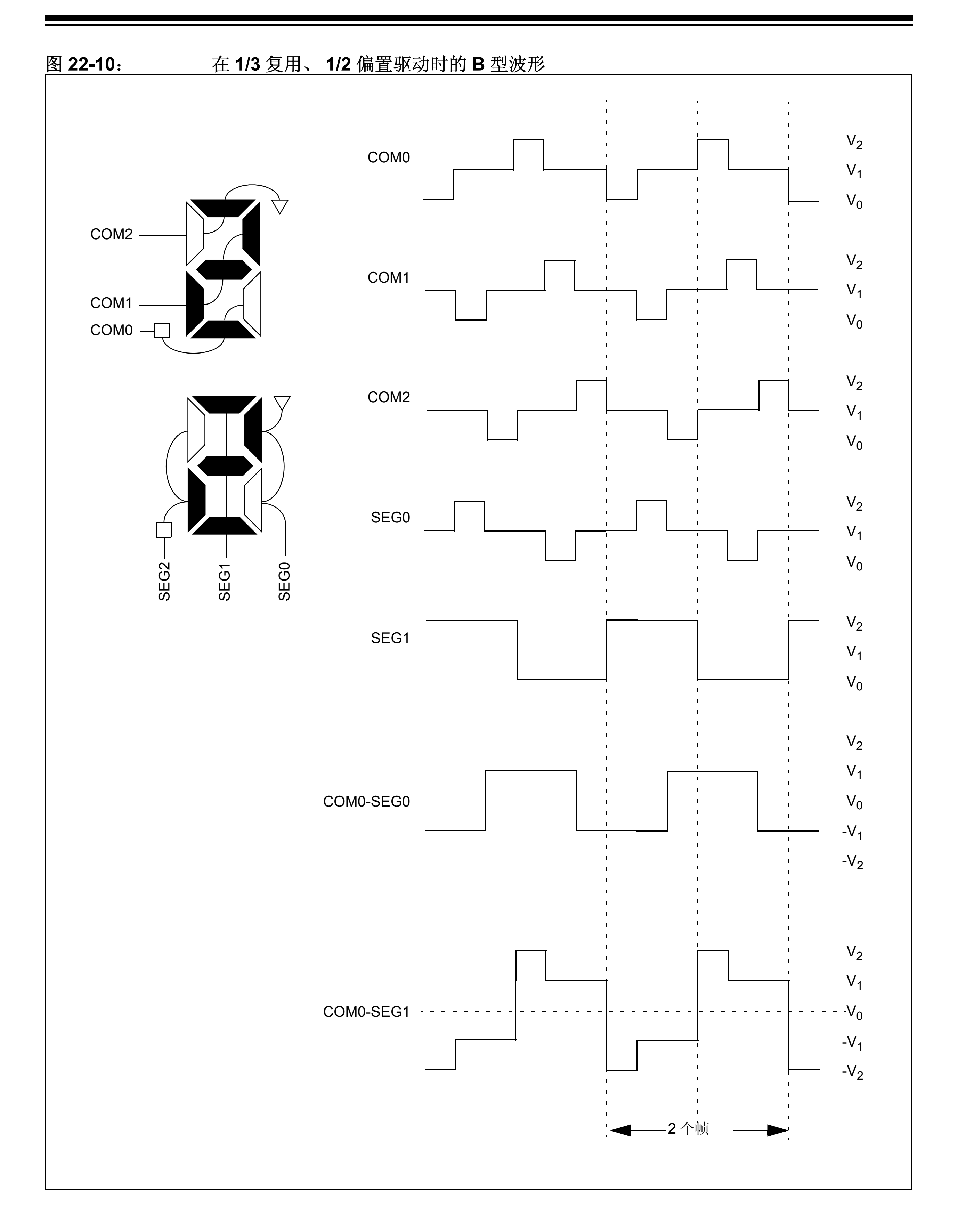
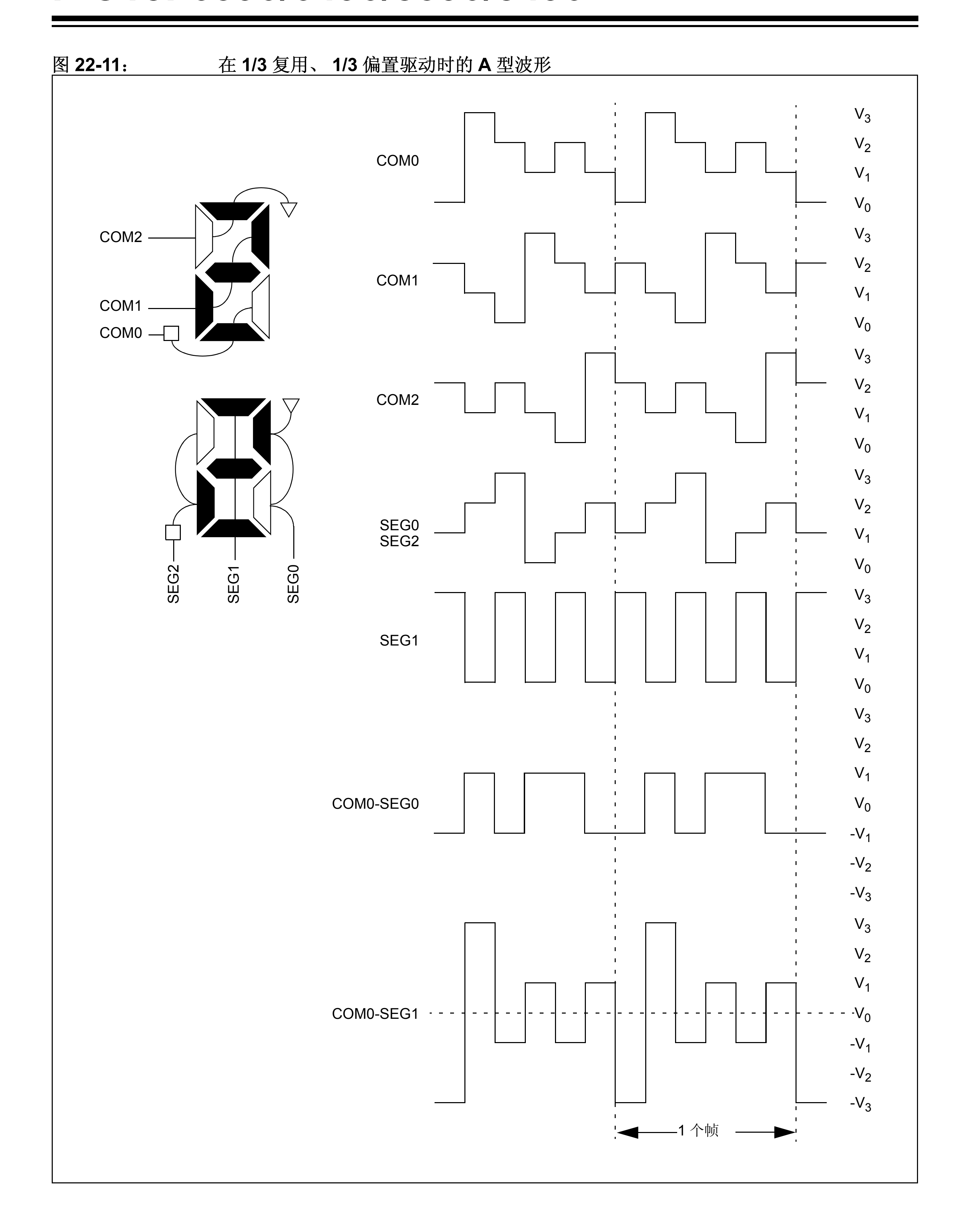


图 22-11： 在 1/3 复用、 1/3 偏置驱动时的 A 型波形



 初稿

图 **22-12**： 在 **1/3** 复用、 **1/3** 偏置驱动时的 **B** 型波形

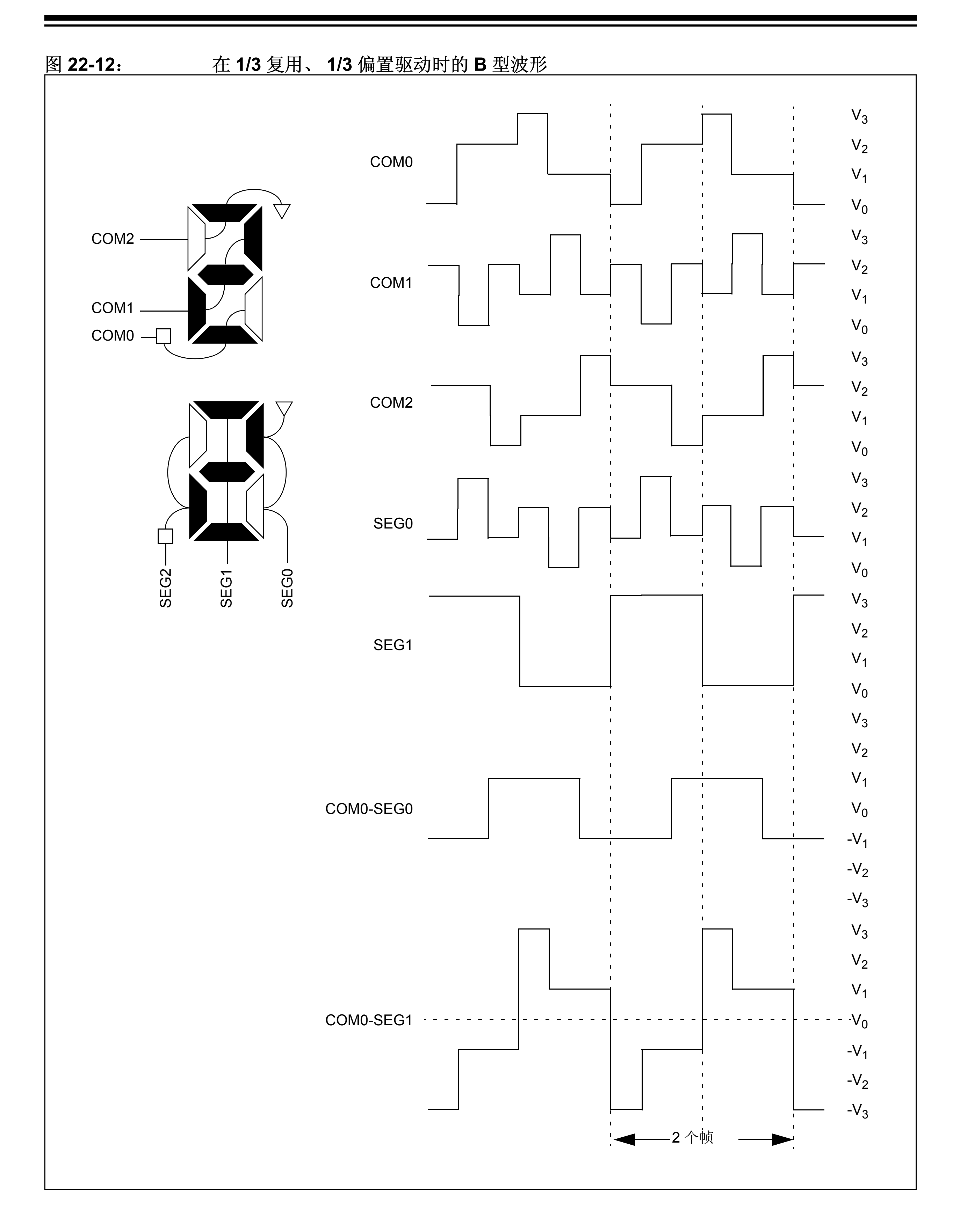
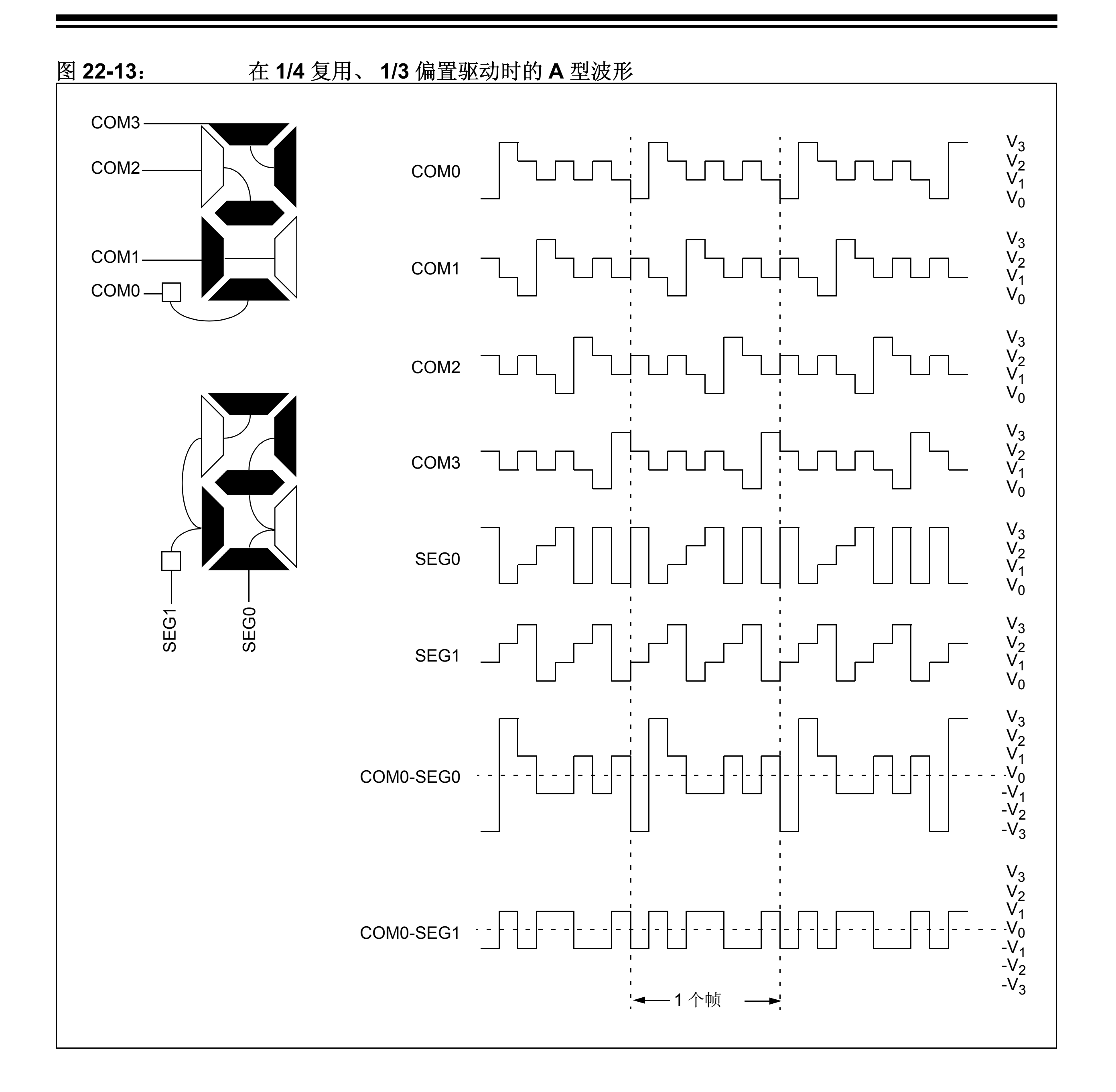
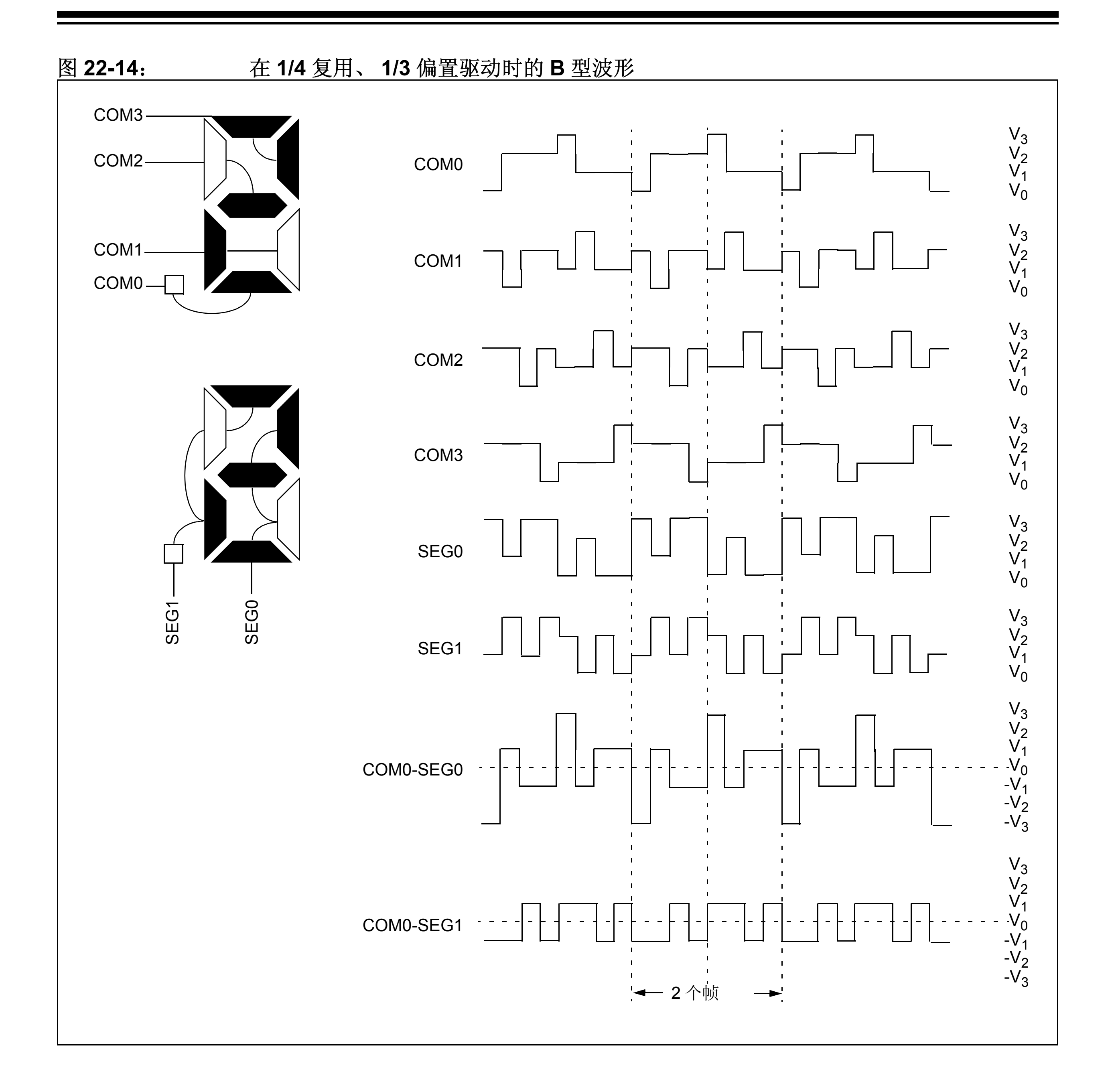


图 22-13： 在 1/4 复用、 1/3 偏置驱动时的 A 型波形



 初稿

图 **22-14**： 在 **1/4** 复用、 **1/3** 偏置驱动时的 **B** 型波形



22.9 LCD 中断

LCD定时发生器提供了一个中断，该中断用于定义LCD的帧时序。它可以在一个新帧开始时写入像素数据，在帧边界处写像素数据可使图像过渡更清晰。该中断还可用于同步 LCD 和外部事件。例如，与外部段驱动器的接口，使更新段数据与 LCD 帧同步。

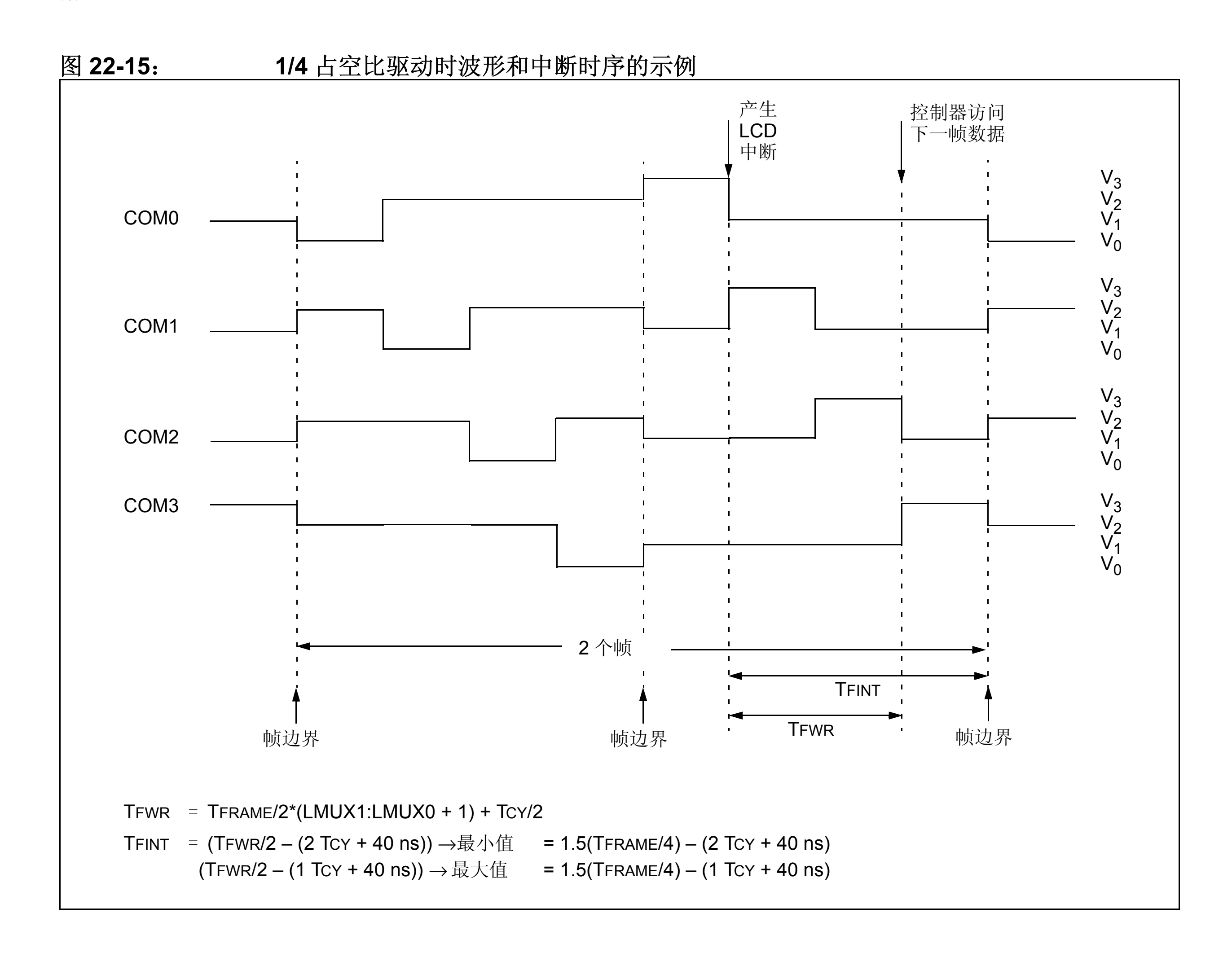
一个新帧开始于COM0公共端信号的起始边界。在LCD控制器完成对帧所需的像素数据访问后，将立即产生中断。中断发生在帧边界（TFINT）前的某一固定时间（如图 22-15 所示）。在中断发生的 TFWR 时间后，LCD 控制器将开始访问下一帧数据。新数据必须在 TFWR 内写入，因为 TFWR 之后，LCD 控制器将开始访问下一帧数据。

当 LCD 驱动器运行 B 型波形且 LMUX1:LMUX0 位不为 00 时，需要处理一些额外问题。由于需要用两帧时间来维持像素上的 DC 电压为零，因此在此期间像素数据要保持不变。一旦像素数据发生改变，奇数帧波形和偶数帧波形不再互补，在面板中会引入一个直流分量。因此，当使用 B 型波形时，用户必须同步 LCD 像素更新，该更新发生在帧中断后的下一帧中。

在 B 型波形时要保证正确的写入时序，中断只能发生在完整的相位间隔内。当禁止写入时，一旦用户试图进行写操作，标志位 WERR （LCDCON<5>）将被置 1。

注：	当选择 A 型波形和选择不带复用 （静态）的 B 型波形时，不会产生中断。

图 22-15： 1/4 占空比驱动时波形和中断时序的示例



 初稿

22.10 在休眠模式下的操作

LCD 模块可以工作在休眠模式下。模式选择由 SLPEN（LCDCON<6>）位控制。置位 SLPEN 将允许 LCD 模块进入休眠模式。清零 SLPEN 位将使模块在休眠模式下继续工作。

如果执行 SLEEP 指令并且当 SLPEN = 1 时，LCD 模块将中止所有的功能，进入极低电流消耗模式。模块将立即停止工作并在段和公共端上输出最小LCD驱动电压，如图 22-16 所示。

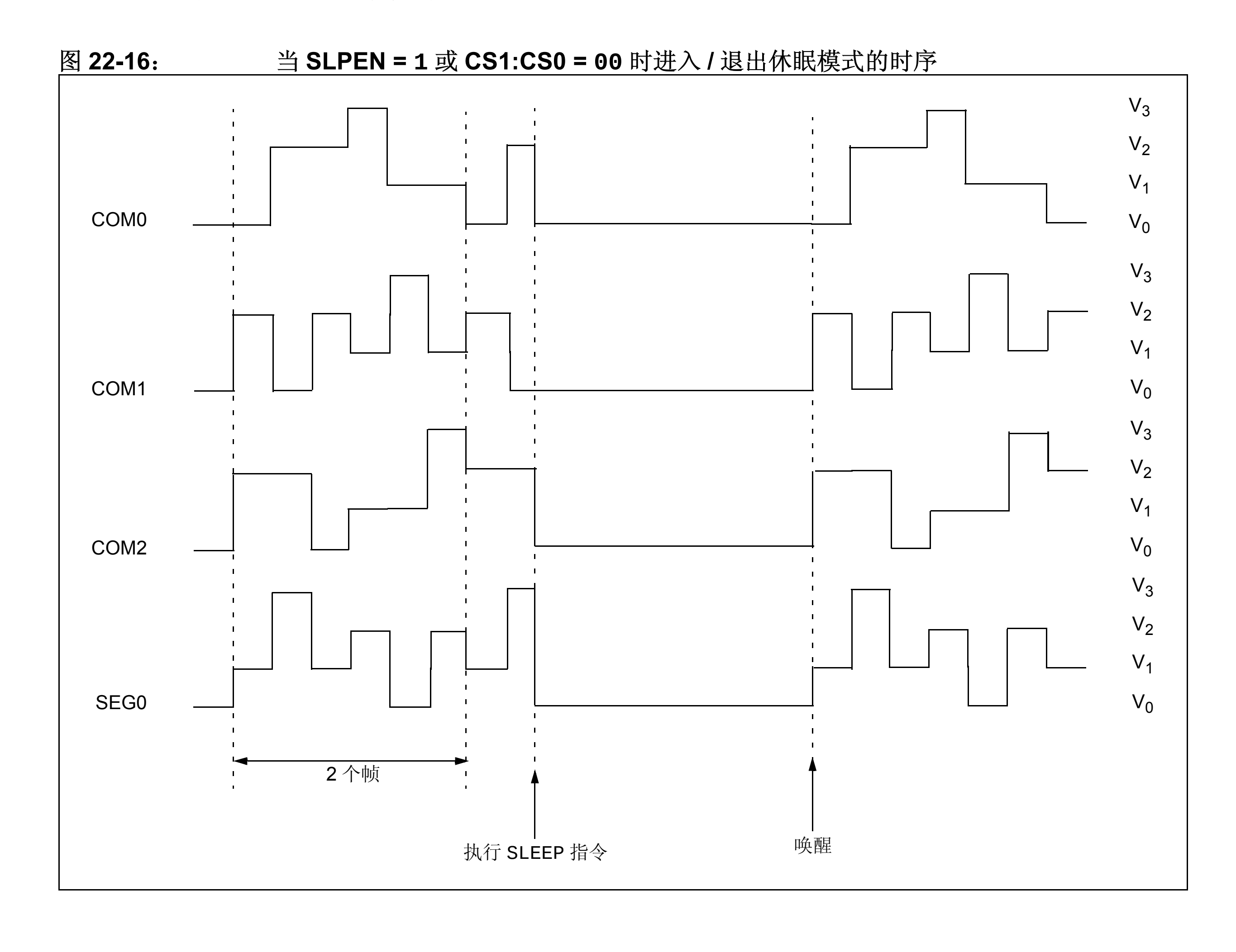
为确保没有直流分量引入面板， SLEEP 指令应紧跟在 LCD 帧边界后执行。可用 LCD 中断判定帧边界。延时的计算请参见**第 22.9 节 "LCD 中断"**内的公式。

如果执行 SLEEP 指令且 SLPEN = 0，模块将继续显示 LCDDATA 寄存器中目前的内容。要使模块在休眠模式下进行工作，时钟源必须为内部 RC 振荡器或 Timer1 外部振荡器。在休眠模式下，LCD 数据不能改变。在此模式下，LCD 模块电流消耗并未降低，然而，器件的整体消耗将因内核和其他外设功能的关闭而降低。

如果使用系统时钟并把模块编程设置为非休眠模式，模块将忽略 SLPEN 位而立即停止操作。段和公共端上输出的为最小 LCD 驱动电压。

注：	在休眠模式下 LCD 模块工作必须选择内部 RC 振荡器或外部 Timer1 振荡器。

图 22-16： 当 SLPEN = 1 或 CS1:CS0 = 00 时进入 / 退出休眠模式的时序



22.11 配置 LCD 模块

以下是配置 LCD 模块的步骤：

1. 使用 LP3:LP0（LCDPS<3:0>）位选择帧时钟预分频比。
2. 使用LCDSEx寄存器把相应的引脚配置为段驱动引脚。
3. 使用 LCDCON 寄存器配置 LCD 模块：
   - 使用 LMUX1:LMUX0 位，配置复用和偏置模式
   - 使用 CS1:CS0 配置时钟源
   - 使用 SLPEN 位配置休眠模式
4. 把像素数据的初始化值写入 LCDDATA0:LCDDATA23 寄存器。
5. 清零 LCD 中断标志位 LCDIF （PIR3<6>），如有必要，置位 LCDIE （PIE3<6>）使能中断。
6. 置位 LCDEN（LCDCON<7>）位，使能 LCD 模块。

表 **22-6**： 与 **LCD** 操作相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
PIR3	—	LCDIF	RC2IF	TX2IF	—	—	—	—	61
PIE3	—	LCDIE	RC2IE	TX2IE	—	—	—	—	61
IPR3	—	LCDIP	RC2IP	TX2IP	—	—	—	—	61
RCON	IPEN	SBOREN	—	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$	60
LCDDATA23(1)	S47C3	S46C3	S45C3	S44C3	S43C3	S42C3	S41C3	S40C3	63
LCDDATA22(1)	S39C3	S38C3	S37C3	S36C3	S35C3	S34C3	S33C3	S32C3	63
LCDDATA21	S31C3	S30C3	S29C3	S28C3	S27C3	S26C3	S25C3	S24C3	63
LCDDATA20	S23C3	S22C3	S21C3	S20C3	S19C3	S18C3	S17C3	S16C3	63
LCDDATA19	S15C3	S14C3	S13C3	S12C3	S11C3	S10C3	S09C3	S08C3	63
LCDDATA18	S07C3	S06C3	S05C3	S04C3	S03C3	S02C3	S01C3	S00C3	63
LCDDATA17(1)	S47C2	S46C2	S45C2	S44C2	S43C2	S42C2	S41C2	S40C2	63
LCDDATA16(1)	S39C2	S38C2	S37C2	S36C2	S35C2	S34C2	S33C2	S32C2	63
LCDDATA15	S31C2	S30C2	S29C2	S28C2	S27C2	S26C2	S25C2	S24C2	63
LCDDATA14	S23C2	S22C2	S21C2	S20C2	S19C2	S18C2	S17C2	S16C2	63
LCDDATA13	S15C2	S14C2	S13C2	S12C2	S11C2	S10C2	S09C2	S08C2	63
LCDDATA12	S07C2	S06C2	S05C2	S04C2	S03C2	S02C2	S01C2	S00C2	63
LCDDATA11(1)	S47C1	S46C1	S45C1	S44C1	S43C1	S42C1	S41C1	S40C1	63
LCDDATA10(1)	S39C1	S38C1	S37C1	S36C1	S35C1	S34C1	S33C1	S32C1	63
LCDDATA9	S31C1	S30C1	S29C1	S28C1	S27C1	S26C1	S25C1	S24C1	63
LCDDATA8	S23C1	S22C1	S21C1	S20C1	S19C1	S18C1	S17C1	S16C1	63
LCDDATA7	S15C1	S14C1	S13C1	S12C1	S11C1	S10C1	S09C1	S08C1	63
LCDDATA6	S07C1	S06C1	S05C1	S04C1	S03C1	S02C1	S01C1	S00C1	63
LCDDATA5(1)	S47C0	S46C0	S45C0	S44C0	S43C0	S42C0	S41C0	S40C0	63
LCDDATA4(1)	S39C0	S38C0	S37C0	S36C0	S35C0	S34C0	S33C0	S32C0	63
LCDDATA3	S31C0	S30C0	S29C0	S28C0	S27C0	S26C0	S25C0	S24C0	63
LCDDATA2	S23C0	S22C0	S21C0	S20C0	S19C0	S18C0	S17C0	S16C0	63
LCDDATA1	S15C0	S14C0	S13C0	S12C0	S11C0	S10C0	S09C0	S08C0	63
LCDDATA0	S07C0	S06C0	S05C0	S04C0	S03C0	S02C0	S01C0	S00C0	63
LCDSE5(2)	SE47	SE46	SE45	SE44	SE43	SE42	SE41	SE40	63
LCDSE4(2)	SE39	SE38	SE37	SE36	SE35	SE34	SE33	SE32	64
LCDSE3	SE31	SE30	SE29	SE28	SE27	SE26	SE25	SE24	64
LCDSE2	SE23	SE22	SE21	SE20	SE19	SE18	SE17	SE16	64
LCDSE1	SE15	SE14	SE13	SE12	SE11	SE10	SE9	SE8	64
LCDSE0	SE7	SE6	SE5	SE4	SE3	SE2	SE1	SE0	64
LCDCON	LCDEN	SLPEN	WERR	—	CS1	CS0	LMUX1	LMUX0	64
LCDPS	WFT	BIASMD	LCDA	WA	LP3	LP2	LP1	LP0	64

图注： — = 未用位，读为 0。 LCD 操作未使用阴影单元。

注 **1**： 64 引脚器件中包含这些寄存器但未使用，可用作通用数据 RAM。

2： 64 引脚器件中不包含这些寄存器。

 初稿

注：

23.0 CPU 的特殊功能

PIC18F6390/6490/8390/8490 器件具有几项特殊的功能旨在最大限度地提高系统可靠性，并通过减少外部元件将成本降至最低。这些功能包括：

- 振荡器选择
- 复位：
  - 上电复位 （POR）
  - 上电延迟定时器 （PWRT）
  - 振荡器起振定时器 （OST）
  - 欠压复位 （BOR）
- 中断
- 看门狗定时器 （WDT）
- 故障保护时钟监视器 （Fail-Safe Clock Monitor，FSCM）
- 双速启动
- 代码保护
- ID 单元
- 在线串行编程

要根据具体应用对频率、功耗、精度和成本的要求来选择振荡器。在**第 2.0 节 "振荡器配置"**中详细讨论了所有的选项。

在本数据手册的前面几章中已完整地讨论了器件的复位和中断。

除了为复位提供了上电延迟定时器和振荡器起振定时器之外，PIC18F6390/6490/8390/8490 器件还提供了一个看门狗定时器，该定时器可配置成永久使能或用软件控制 （如果使能位被禁止的话）。

器件自带的内部 RC 振荡器还提供了故障保护时钟监视器 （FSCM）和双速启动这两个额外的功能。FSCM 对外设时钟进行后台监视，并在外设时钟发生故障时自动切换时钟源。双速启动使得几乎可在起振发生那一刻立即执行代码，同时主时钟源继续其起振延时。

通过设置相应的配置寄存器位可以使能和配置所有这些功能。

23.1 配置位

可以通过对配置位编程 （读为 0）或不编程 （读为 1）来选择不同的器件配置。这些配置位被映射到程序存储器以 300000h 开始的单元中。

用户会注意到地址300000h超出了用户程序存储器空间范围。事实上，它属于配置存储器空间 （300000h — 3FFFFFh），这一空间仅能通过表读进行访问。

表 23-1： 配置位和器件 ID

寄存器名称		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	缺省 / 未编程值
300001h	CONFIG1H	IESO	FCMEN	—	—	FOSC3	FOSC2	FOSC1	FOSC0	00-- 0111
300002h	CONFIG2L	—	—	—	BORV1	BORV0	BOREN1	BOREN0	PWRTEN	---1 1111
300003h	CONFIG2H	—	—	—	WDTPS3	WDTPS2	WDTPS1	WDTPS0	WDTEN	---1 1111
300005h	CONFIG3H	MCLRE	—	—	—	—	LPT1OSC	—	CCP2MX	1--- -0-1
300006h	CONFIG4L	DEBUG	XINST	—	—	—	—	—	STVREN	10-- ---1
300008h	CONFIG5L	—	—	—	—	—	—	—	CP	---- ---1
3FFFFEh	DEVID1	DEV2	DEV1	DEV0	REV4	REV3	REV2	REV1	REV0	xxxx xxxx(1)
3FFFFFh	DEVID2	DEV10	DEV9	DEV8	DEV7	DEV6	DEV5	DEV4	DEV3	0000 xxxx(1)

图注： x = 未知， u = 不变， — = 未用， q = 取值视情况而定。
阴影单元未用，读为 0。

注 1： 关于 DEVID 的值，请参见寄存器 23-7。 DEVID 寄存器为只读寄存器，用户不能对其进行编程。

寄存器 23-1： CONFIG1H：配置寄存器 1 的高字节 （字节地址 = 300001h）

R/P-0	R/P-0	U-0	U-0	R/P-0	R/P-1	R/P-1	R/P-1
IESO	FCMEN	—	—	FOSC3	FOSC2	FOSC1	FOSC0
bit 7							bit 0

bit 7 **IESO**：内部 / 外部振荡器切换位
1 = 使能振荡器切换模式
0 = 禁止振荡器切换模式

bit 6 **FCMEN**：故障保护时钟监视器使能位
1 = 使能故障保护时钟监视器
0 = 禁止故障保护时钟监视器

bit 5-4 未用位：读为 0

bit 3-0 **FOSC3:FOSC0**：振荡器选择位
11xx = 外部 RC 振荡器， RA6 用作 CLKO 引脚
101x = 外部 RC 振荡器， RA6 用作 CLKO 引脚
1001 = 内部振荡器电路， RA6 用作 CLKO 引脚， RA7 用作端口引脚
1000 = 内部振荡器电路， RA6 和 RA7 均用作端口引脚
0111 = 外部 RC 振荡器， RA6 用作端口引脚
0110 = PLL 使能的 HS 振荡器 （时钟频率 =4 x FOSC1）
0101 = EC 振荡器， RA6 用作端口引脚
0100 = EC 振荡器， RA6 用作 CLKO 引脚
0011 = 外部 RC 振荡器， RA6 用作 CLKO 引脚
0010 = HS 振荡器
0001 = XT 振荡器
0000 = LP 振荡器

图注：		
R = 可读位	P = 可编程位	U = 未用位，读为 0
-n = 未对器件编程时的值		u = 编程后状态不变

寄存器 23-2： CONFIG2L：配置寄存器 2 的低字节 （字节地址 = 300002h）

U-0	U-0	U-0	R/P-1	R/P-1	R/P-1	R/P-1	R/P-1
—	—	—	BORV1	BORV0	BOREN1(1)	BOREN0(1)	$\overline{\text{PWRTEN}}$(1)
bit 7							bit 0

bit 7-5 未用位：读为 0

bit 4-3 **BORV1:BORV0**：欠压复位门限电压位
11 = VBOR 置为 2.1V
10 = VBOR 置为 2.8V
01 = VBOR 置为 4.3V
00 = VBOR 置为 4.6V

bit 2-1 **BOREN1:BOREN0** 欠压复位使能位 (1)
11 = 由硬件使能欠压复位 （禁止 SBOREN）
10 = 由硬件使能欠压复位，休眠模式下被禁止 （禁止 SBOREN）
10 = 由软件使能和控制欠压复位 （使能 SBOREN）
10 = 禁止使用硬件或软件使能欠压复位

bit 0 **$\overline{\text{PWRTEN}}$**：上电延迟定时器 （PWRT）使能位 (1)
1 = 禁止 PWRT
0 = 使能 PWRT

注 1：上电延时定时器与欠压复位是相互独立的，可以分别控制两者的操作。

图注：		
R = 可读位	P = 可编程位	U = 未用位，读为 0
-n = 未对器件编程时的值		u = 编程后状态不变

寄存器 23-3： CONFIG2H：配置寄存器 2 的高字节 （字节地址 = 300003h）

U-0	U-0	U-0	R/P-1	R/P-1	R/P-1	R/P-1	R/P-1
—	—	—	WDTPS3	WDTPS2	WDTPS1	WDTPS0	WDTEN
bit 7							bit 0

bit 7-5 未用位：读为 0

bit 4-1 **WDTPS3:WDTPS0**：看门狗定时器后分频比选择位
1111 = 1:32,768
1110 = 1:16,384
1101 = 1:8,192
1100 = 1:4,096
1011 = 1:2,048
1010 = 1:1,024
1001 = 1:512
1000 = 1:256
0111 = 1:128
0110 = 1:64
0101 = 1:32
0100 = 1:16
0011 = 1:8
0010 = 1:4
0001 = 1:2
0000 = 1:1

bit 0 **WDTEN**：看门狗定时器使能位
1 = 使能 WDT
0 = 禁止 WDT （由 SWDTEN 控制）

图注：		
R = 可读位	P = 可编程位	U = 未用位，读为 0
-n = 未对器件编程时的值		u = 编程后状态不变

寄存器 23-4： CONFIG3H：配置寄存器 3 的高字节 （字节地址 = 300005h）

R/P-1	U-0	U-0	U-0	U-0	R/P-0	U-0	R/P-1
MCLRE	—	—	—	—	LPT1OSC	—	CCP2MX
bit 7							bit 0

bit 7 **MCLRE**：$\overline{\text{MCLR}}$ 引脚使能位
1 = 使能 $\overline{\text{MCLR}}$ 引脚；禁止 RG5 输入引脚
0 = 使能 RG5 输入引脚；禁止 $\overline{\text{MCLR}}$

bit 6-3 未用位：读作 0

bit 2 **LPT1OSC**：低功耗 Timer1 振荡器使能位
1 = Timer1 配置为低功耗运行
0 = Timer1 配置为高功耗运行

bit 1 未用位：读为 0

bit 0 **CCP2MX**：CCP2 复用位
1 = CCP2 输入 / 输出与 RC1 复用
0 = CCP2 输入 / 输出与 RE7 复用

图注：		
R = 可读位	P = 可编程位	U = 未用位，读为 0
-n = 未对器件编程时的值		u = 编程后状态不变

寄存器 23-5： CONFIG4L：配置寄存器 4 的低字节 （字节地址 = 300006h）

R/P-1	R/P-0	U-0	U-0	U-0	U-0	U-0	R/P-1
$\overline{\text{DEBUG}}$	XINST	—	—	—	—	—	STVREN
bit 7							bit 0

bit 7 **$\overline{\text{DEBUG}}$**：后台调试器使能位
1 = 禁止后台调试器， RB6 和 RB7 被配置为通用 I/O 引脚
0 = 使能后台调试器， RB6 和 RB7 专用于在线调试

bit 6 **XINST**：扩展指令集使能位
1 = 使能指令集扩展和变址寻址模式
0 = 禁止指令集扩展和变址寻址模式 （传统模式）

bit 5-1 未用位：读为 0

bit 0 **STVREN**：堆栈满 / 下溢复位使能位
1 = 堆栈满 / 下溢导致复位
0 = 堆栈满 / 下溢不会导致复位

图注：		
R = 可读位	C = 可清零位	U = 未用位，读为 0
-n = 未对器件编程时的值		u = 编程后状态不变

寄存器 23-6： CONFIG5L：配置寄存器 5 的低字节 （字节地址 = 300008h）

U-0	U-0	U-0	U-0	U-0	U-0	U-0	R/C-1
—	—	—	—	—	—	—	CP
bit 7							bit 0

bit 7-1 未用位：读为 0

bit 0 **CP**：代码保护位
1 = 程序存储区 （000000-003FFFh）无代码保护
0 = 程序存储区 （000000-003FFFh）有代码保护

图注：		
R = 可读位	C = 可清零位	U = 未用位，读为 0
-n = 未对器件编程时的值		u = 编程后状态不变

寄存器 23-7： PIC18F6390/6490/8390/8490 器件的器件 ID 寄存器 1

R	R	R	R	R	R	R	R
DEV2	DEV1	DEV0	REV4	REV3	REV2	REV1	REV0
bit 7							bit 0

bit 7-5 **DEV2:DEV0：**器件 ID 位
`100` = PIC18F8390/8490
`101` = PIC18F6390/6490

bit 4-0 **REV4:REV0：**版本 ID 位
这些位用于表明器件版本。

图注：		
R = 只读位	P = 可编程位	U = 未用位，读为 0
-n = 未对器件编程时的值		u = 编程后状态不变

寄存器 23-8： PIC18F6390/6490/8390/8490 器件的器件 ID 寄存器 2

R	R	R	R	R	R	R	R
DEV10	DEV9	DEV8	DEV7	DEV6	DEV5	DEV4	DEV3
bit 7							bit 0

bit 7-0 **DEV10:DEV3：**器件 ID 位
这些位与器件 ID 寄存器 1 中的 DEV2:DEV0 一起用于标识器件号。
`0000 0110` = PIC18F6490/8490 器件
`0000 1011` = PIC18F6390/8390 器件

注： DEV10:DEV3 的值可能会用于其他器件。特定器件是通过使用整个 DEV10:DEV0 位序列来标识的。

图注：		
R = 只读位	P = 可编程位	U = 未用位，读为 0
-n = 未对器件编程时的值		u = 编程后状态不变

23.2 看门狗定时器 （WDT）

PIC18F6390/6490/8390/8490 器件的 WDT 是由 INTRC 时钟源驱动的。当使能 WDT 时，时钟源也将同时使能。WDT 超时溢出周期的标称值为 4 ms，其稳定性与 INTRC 振荡器相同。

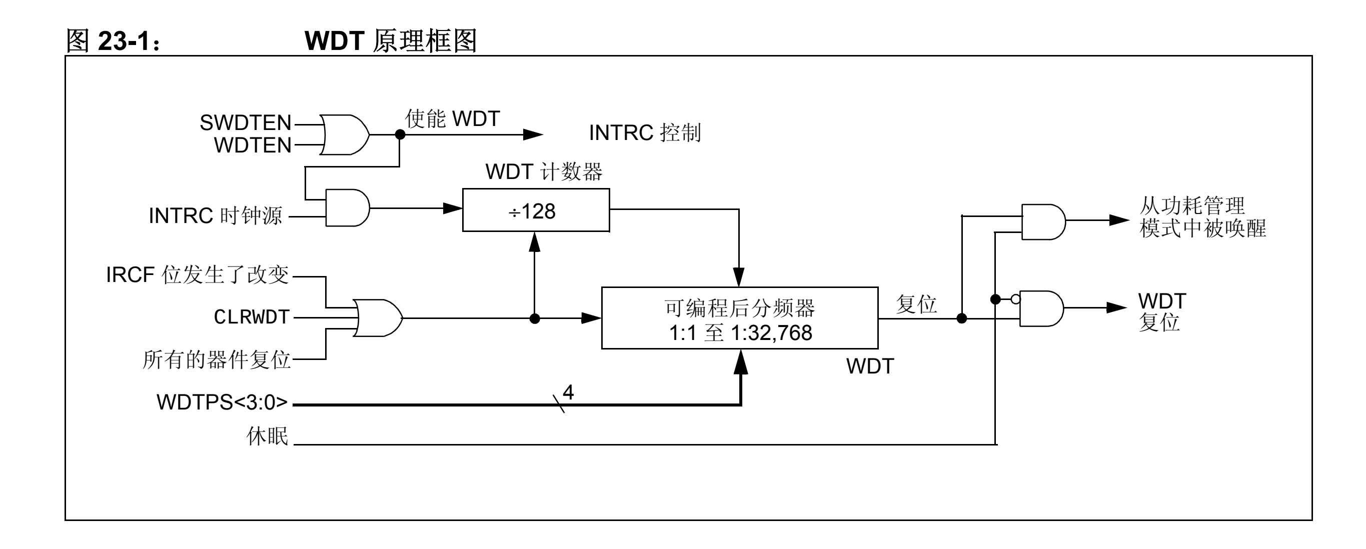
4 ms 的 WDT 超时溢出周期将与 16 位后分频器的值相乘来得到更长的时间周期。通过配置寄存器 2H 来控制一个多路开关以对 WDT 后分频器的输出进行选择。因此可获得的超时溢出周期范围为 4 ms 至 134.2 秒（2.24 分钟）。当发生以下任一事件时，WDT 和后分频器将被清零，这些事件包括：执行 SLEEP 或 CLRWDT 指令、IRCF 位 （OSCCON<6:4>）发生了改变或发生时钟故障。

注	
1：	当执行 CLRWDT 和 SLEEP 指令时， WDT 和后分频器的计数值将被清零。
2：	更改 IRCF 位 （OSCCON<6:4>）的设置会清零 WDT 和后分频器的计数值。

23.2.1 控制寄存器

寄存器 23-9 表示 WDTCON 寄存器。它是一个可读写的寄存器并包含一个控制位，当 WDT 被配置为禁止时，该控制位允许使用软件来控制 WDT。

图 23-1： WDT 原理框图



寄存器 23-9： **WDTCON：看门狗定时器控制寄存器**

U-0	U-0	U-0	U-0	U-0	U-0	U-0	R/W-0
—	—	—	—	—	—	—	SWDTEN
bit 7							bit 0

bit 7-1 未用位：读为 0

bit 0 **SWDTEN：**由软件控制的看门狗定时器使能位
1 = 打开看门狗定时器
0 = 关闭看门狗定时器

注： 当使能 WDTEN 位时该位不起作用。

图注：	
R = 可读位	W = 可写位
U = 未用位，读为 0	-n = 上电复位时的值

表 23-2： 看门狗定时器寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
RCON	IPEN	SBOREN	—	$\overline{\text{RI}}$	$\overline{\text{TO}}$	$\overline{\text{PD}}$	$\overline{\text{POR}}$	$\overline{\text{BOR}}$	61
WDTCON	—	—	—	—	—	—	—	SWDTEN	60

图注： — = 未用位，读为 0。看门狗定时器不使用阴影单元。

23.3 双速启动

双速启动功能允许单片机在主时钟源稳定之前使用 INTRC 振荡器作为时钟源，从而帮助器件最大限度地缩短从振荡器起振到代码执行之间的延时。通过将 IESO 配置位置 1 可使能该功能。

仅当主振荡器模式为 LP、XT、HS 或 HSPLL（基于晶振的模式）时才可使用双速启动。其他时钟源不需要 OST 起振延时；对于这些时钟源，应禁止双速启动。

当使能双速启动时，在上电延时定时器发生超时（使能上电复位）后，器件复位或从休眠模式中被唤醒，此时器件将被配置成使用内部振荡电路作为时钟源。使得在主振荡器起振、OST 运行的同时，代码开始执行。一旦 OST 超时，器件就自动换到 PRI_RUN 模式。

因为复位事件会清零 OSCCON 寄存器，所以在复位发生后最初是不能使用 INTOSC（或后分频器）时钟源的；但是可以直接在基本频率上使用 INTRC 时钟源。为了在唤醒器件时使用速度更快的时钟，通过在复位发生后立即设置 IRCF2:IRCF0，可以选择 INTOSC 或后分频器时钟源以提供更快的时钟速度。对于从休眠模式唤醒的情况，可以在进入休眠模式前设置 IRCF2:IRCF0 来选择 INTOSC 或后分频器时钟源。

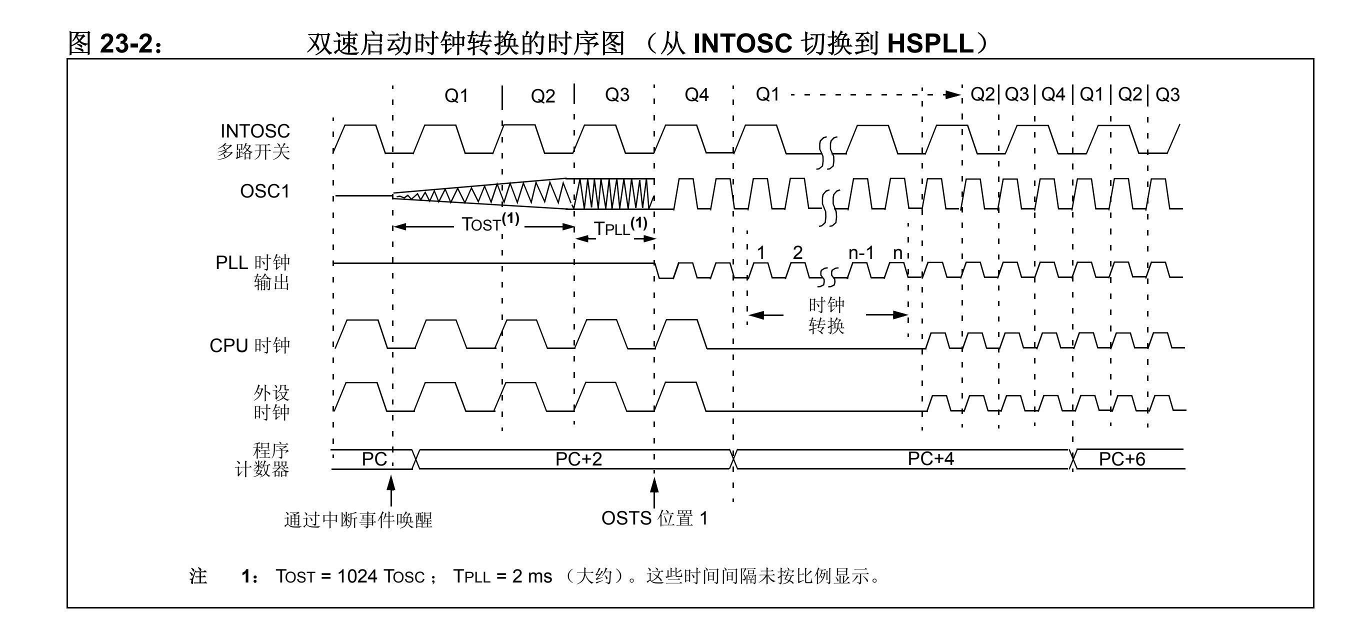
在其他功耗管理模式下，不使用双速启动。器件将使用当前选定的时钟源直到主时钟源可用为止。IESO 位的设置被忽略。

23.3.1 使用双速启动时的注意事项

当在双速启动模式中使用 INTRC 振荡器时，器件仍将遵守进入功耗管理模式（包括执行 `sleep` 指令）的正常指令顺序（见**第 3.1.2 节“进入功耗管理模式”**）。实际上，这意味着在 OST 超时前用户代码可以改变 SCS1:SCS0 位的设置或执行 SLEEP 指令。这就使应用程序能短暂地唤醒器件，执行“日常事务”，并在器件开始使用主时钟源前返回休眠状态。

用户代码还能通过检查 OSTS 位（OSCCON<3>）的状态来确定主时钟源是否正在为系统提供时钟。若该位置 1，则表示主振荡器正在为系统提供时钟。否则，表示当器件复位或从休眠模式被唤醒期间由内部振荡器电路为系统提供时钟。

图 23-2： 双速启动时钟转换的时序图（从 **INTOSC** 切换到 **HSPLL**）



注 1：T_{OST} = 1024 T_{OSC}；T_{PLL} = 2 ms（大约）。这些时间间隔未按比例显示。

23.4 故障保护时钟监视器

故障保护时钟监视器（FSCM）可使单片机在外部时钟发生故障时，自动地将系统时钟切换到内部振荡器电路以保证器件能继续运行。将 FCMEN 配置位置 1 可使能 FSCM 功能。

当使能 FSCM 时， INTRC 振荡器将一直保持运行以监视外设时钟，并且在外设时钟发生故障时作为备用时钟。时钟监视（如图 23-3 所示）通过创建一个采样时钟信号实现，该信号为 INTRC 输出的 64 分频。这样就使得 FSCM 采样时钟脉冲之间有充足的时间间隔，从而保证在此期间至少有一个外设时钟沿出现。外设时钟和采样时钟作为时钟监视器锁存器（CM）的输入。CM 在系统时钟源的下降沿被置 1，在采样时钟的上升沿被清零。

图 23-3： FSCM 框图

在采样时钟的下降沿检测外部时钟故障。如果在出现采样时钟的下降沿时， CM 仍置 1，就表示检测到外部时钟故障（图 23-4）。这将引发以下事件：

- 通过将 OSCFIF （PIR2<7>）置 1，由 FSCM 产生振荡器故障中断；
- 器件时钟源切换为内部振荡器电路（OSCCON 不会被更新，因此无法显示当前时钟源——这就是故障保护状态）；并且
- WDT 复位。

切换过程中，对于定时要求较高的应用，内部振荡器电路的后分频频率可能不够稳定。在这些情况下，最好选择另一种时钟配置并进入其他功耗管理模式。可以尝试部分恢复或执行安全关机。请参见**第 3.1.2 节 “进入功耗管理模式”**和**第 23.3.1 节“使用双速启动时的注意事项”**了解更多详细信息。

为了在唤醒器件时可以使用速度更快的时钟，可以选择 INTOSC 或后分频器时钟源以提供更快的时钟速度，这可以通过在复位发生后立即设置IRCF2:IRCF0实现。对于从休眠模式唤醒的情况，可以在进入休眠模式前设置 IRCF2:IRCF0 来选择 INTOSC 或后分频器时钟源。

FSCM 只能检测出主时钟源或辅助时钟源的故障。如果内部振荡器电路发生故障，将不会被检测到，当然也不可能采取任何措施。

23.4.1 FSCM 和看门狗定时器

FSCM 和 WDT 均以 INTRC 振荡器作为时钟源。由于 WDT 使用独立的分频器和计数器，使能 FSCM 时，禁止 WDT 对 INTRC 振荡器的运行没有影响。

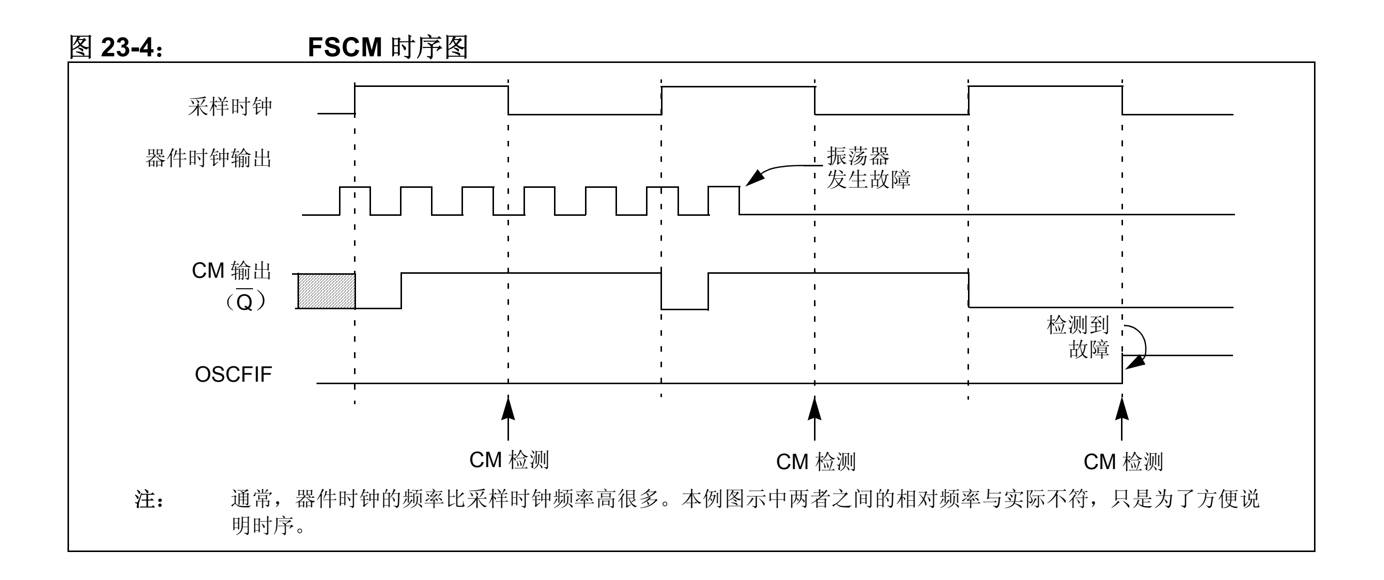
如前所述，当发生时钟故障时，时钟源将切换到 INTOSC 时钟源。根据由 IRCF2:IRCF0 位选择的频率的不同，代码执行速度也会相应发生很大的变化。如果使能 WDT 的时候使用的是最小的预分频值，时钟速度的下降将引起 WDT 超时，随后使器件复位。由于这个原因，故障保护事件也会使 WDT 和后分频器复位，使 WDT 从执行速度发生变化那一刻起开始重新计数，从而避免发生错误超时。

23.4.2 退出故障保护运行模式

器件复位或进入功耗管理模式均可结束故障保护状态。发生复位时，控制器启动在配置寄存器 1H 中指定的主时钟源（伴有如 OST 或 PLL 定时器等所需的起振延时）。 INTOSC 复用器将在主时钟源就绪之前提供系统时钟（类似于双速启动）。当主时钟源可用时，系统时钟源将切换为主时钟（OSCCON 寄存器中的 OSTS 位将置 1，表明当前使用的是主时钟源）。然后，故障保护时钟监视器恢复对外设时钟的监视。

在启动期间，主时钟源可能永远不能就绪。在这种情况下，器件运行将以 INTOSC 复用器作为时钟源。OSCCON 寄存器将保持复位状态直到进入功耗管理模式为止。

图 23-4： FSCM 时序图



注： 通常，器件时钟的频率比采样时钟频率高很多。本例图示中两者之间的相对频率与实际不符，只是为了方便说明时序。

23.4.3 功耗管理模式下的 FSCM 中断

进入功耗管理模式时，时钟多路开关选择由 OSCCON 寄存器选定的时钟源。在该模式下将恢复对功耗管理时钟源的故障保护监视。

如果在功耗管理运行期间发生了振荡器故障，接下来的操作取决于是否使能了振荡器故障中断。如果使能（OSCFIF = 1），代码执行将以 INTOSC 复用器作为时钟源，并且不会自动转回到发生故障的时钟源。

如果禁止了该中断，当发生振荡器故障时，器件将不会退出功耗管理模式。相反，器件将继续像以前一样运行，但是将以 INTOSC 复用器作为时钟源。当处于空闲模式时，振荡器故障所导致的中断将使 CPU 以 INTOSC 复用器作为系统时钟源。

23.4.4 POR 或从休眠中唤醒

FSCM 在器件退出上电复位（POR）或低功耗休眠模式后开始检测振荡器故障。当系统主时钟为 EC、RC 或 INTRC 模式时，监视会在这些事件发生后立即开始。

当振荡器模式（如 HS、HSPLL、LP 或 XT）使用了晶振或谐振器时，情况会有些不同。由于这类振荡器需要的起振时间可能比 FCSM 采样时钟的周期长很多，因此可能会检测到假的时钟故障。为了避免这种情况，在此类模式中，内部振荡器电路会被自动配置为器件时钟并一直工作直到主时钟稳定下来为止（OST 和 PLL 定时器发生超时）。这与双速启动模式相同。一旦主时钟稳定下来， INTRC 就将重新作为 FSCM 时钟源。

注：	相同的逻辑用于防止在 POR 或从休眠状态唤醒时发生错误的中断，同样也将阻止随后对振荡器故障的检测。通过监视 OSTS 位，并使用定时程序来确定振荡器起振时间是否过长，可避免这个问题。即使如此，在检测到振荡器故障时也不会将振荡器故障中断标志位置 1。

正如**第 23.3.1 节 “使用双速启动时的注意事项”**中所述，在等待系统主时钟稳定的过程中，可以选择另一种时钟配置和另一种功耗管理模式。当选择了新的功耗管理模式时，主时钟将被禁止。

23.5 程序校验和代码保护

PIC18F6390/6490/8390/8490 闪存器件的整个代码保护结构与以往的 PIC18 器件有所不同。

对于 PIC18F6X90/8X90 系列中的所有器件，用户程序存储器仅由一个存储区组成。图 23-5 显示了各个器件的程序存储器构成。对该存储区的代码保护由 CP 位（CONFIG5L<0>）控制。CP 位阻止对整个程序存储器空间进行外部读写，但对于代码正常执行没有直接影响。

23.5.1 读程序存储器和其他单元

使用表读指令可以读取程序存储器中任何单元的内容，也可以读取器件 ID 和配置寄存器的内容。

23.5.2 配置寄存器保护

配置寄存器只能使用外部编程器通过 ICSP 写入。配置寄存器没有与之相关的独立保护位。

图 23-5： PIC18F6390/6490/8390/8490 的受代码保护的程序存储器

存储容量 / 器件 8 KB （PIC18F6390/8390）	寻址范围	16 KB （PIC18F6490/8490）	寻址范围	存储区的 代码保护受控于： CP, EBTR
程序存储区	000000h 001FFFh	程序存储区	000000h 003FFFh	
未用 读为 0	002000h 1FFFFFh	未用 读为 0	004000h 1FFFFFh	（未用的存储器空间）

表 23-3： 代码保护寄存器汇总

寄存器名称		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
300008h	CONFIG5L	—	—	—	—	—	—	—	CP

图注： 未使用阴影单元。

23.6 ID 单元

有 8 个存储器单元（200000h-200007h）被指定为 ID 单元，供用户存储校验和或其他代码标识。在执行程序时可通过 TBLRD 指令读取这些单元；在编程 / 验证时，可对这些地址单元进行读写操作。当器件有代码保护时，也可读取 ID 单元。

23.7 在线串行编程

PIC18F6390/6490/8390/8490 单片机可以在最终的应用电路中进行串行编程。只需要 5 根线即可实现这一操作，其中时钟线、数据线各一根，其余 3 根分别是电源线、接地线和编程电压线。这允许用户在制造电路板时使用未编程器件，仅在产品交付之前才对单片机进行编程，从而可以使用最新版本的固件或者定制固件。

23.8 在线调试器

将 DEBUG 配置位清 0，可使能在线调试功能，该功能允许使用 MPLAB® IDE 进行一些简单的调试。当使能了单片机的这项功能时，有些资源就不再是通用的了。表 23-4 显示了后台调试器所需的资源。

表 23-4： 调试器资源

I/O 引脚：	RB6， RB7
堆栈：	2 级
程序存储器：	512 字节
数据存储器：	10 字节

要使用单片机的在线调试器功能，在设计时就必须实现在线编程模块与 $\overline{MCLR}/V_{PP}$、V_{DD}、V_{SS}、RB7 和 RB6 的正确连接，从而为 Microchip 或第三方开发工具公司提供的在线调试器模块提供交互的接口。

注：

24.0 指令集综述

PIC18F6390/6490/8390/8490 器件具有一个含有 75 个 PIC18 内核指令的标准指令集，和一个含有优化递归或软件堆栈代码的 8 个新指令的扩展指令集。本章后面的部分将讨论扩展指令集。

24.1 标准指令集

标准的 PIC18C 指令集与以前的 PICmicro® 指令集相比，添加了很多增强功能，并保持了易于从其他 PICmicro 指令集移植的特点。大部分指令为单字指令（16 位），只有 4 个指令是双字指令。

每个单字指令都是一个 16 位字，由操作码（指明指令类型）和一个或多个操作数（指定指令操作）组成。

整个指令集具有高度的正交性，可以分为以下 4 种基本类型：

- **字节**操作类指令
- **位**操作类指令
- **立即数**操作类指令
- **控制**操作类指令

表 24-2为PIC18指令集汇总。表 24-1给出了操作码字段的说明。

大部分**字节操作类**的指令都含有三种操作数：

1. 文件寄存器（由 “f” 指定）
2. 保存结果的目标寄存器（由 “d” 指定）
3. 被访问存储器（由 “a” 指定）

文件寄存器标识符 “f” 指定了指令将会使用哪一个文件寄存器。目标寄存器标识符 “d” 指定了操作结果的存放位置。如果 “d” 为 0，操作结果存入 WREG 寄存器中；如果 “d” 为 1，操作结果存入指令指定的文件寄存器中。

所有**位操作类**指令都含有三种操作数：

1. 文件寄存器（由 “f” 指定）
2. 文件寄存器中的位（由 “b” 指定）
3. 被访问存储器（由 “a” 指定）

位域标识符 “b” 选择操作所影响的位的编号，而文件寄存器标识符 “f” 则代表这些位所在的寄存器编号。

立即数操作指令使用以下操作数：

- 要装入文件寄存器中的立即数（由 “k” 指定）
- 要装入立即数的 FSR 寄存器（由 “f” 指定）
- 不需要操作数（由 “—” 指定）

控制类指令可以使用以下操作数：

- 程序存储器地址（由 “n” 指定）
- CALL 或 RETURN 指令的模式（由 “s” 指定）
- 表读和表写指令的模式（由 “m” 指定）
- 不需要操作数（由 “—” 指定）

除了 4 个双字指令外，所有的指令都是单字指令。双字指令将所需的信息保存在 32 位中。第二个字的高 4 位都是1。如果第二个字作为一条指令执行，它会执行NOP 指令。

除非条件测试结果为true或者指令执行改变了程序计数器的值，否则执行所有的单字指令都只需要一个指令周期。对于上述两种特殊情况，指令执行需要两个指令周期，在第二个指令周期中执行一条 NOP 指令。

执行双字指令需要两个指令周期。

每个指令周期由 4 个振荡器周期组成。因此，对于频率为 4 MHz 的振荡器，其正常的指令执行时间为 1 µs。如果条件测试为 true 或指令执行改变了程序计数器的值，则该指令的执行时间为 2 µs。双字跳转指令（如果为 true）的执行则需要 3 µs。

图 24-1 给出了指令的几种通用格式。所有示例均使用 “nnh” 来表示十六进制数。

指令集汇总（见表 24-2）列出了可被 Microchip MPASM™ 汇编器识别的标准指令。

第 **24.1.1 节 “标准指令集”** 中对每个指令进行了介绍。

表 **24-1**： 操作码字段说明

字段	说明
a	快速操作 RAM 位 a = 0：快速操作 RAM 内的 RAM 单元 （BSR 寄存器被忽略） a = 1：由 BSR 寄存器指定的 RAM 快速操作存储区
bbb	8 位文件寄存器内的位地址 （0 到 7）。
BSR	存储区选择寄存器。用于选择当前的 RAM 存储区。
C、 DC、 Z、 OV 和 N	ALU 状态位：进位标志位、辅助进位标志位、全零标志位、溢出标志位和负标志位。
d	目标寄存器选择位 d = 0：结果保存至 WREG 寄存器 d = 1：结果保存至文件寄存器 f。
dest	目标寄存器：可以是 WREG 寄存器或指定的寄存器地址。
f	8 位寄存器地址 （00h 到 FFh），或 2 位 FSR 标识符 （0h 到 3h）。
f_s	12 位寄存器地址 （000h 到 FFh）。这是源地址。
f_d	12 位寄存器地址 （000h 到 FFFh）。这是目标地址。
GIE	全局中断使能位。
k	立即数、常数或者标号 （可能是 8 位、 12 位或 20 位的值）
标号	标号名称
mm * *+ *- +*	表读和表写指令的 TBLPTR 寄存器模式。只与表读和表写指令一起使用： 不改变寄存器 （如用于表读和表写的 TBLPTR）。 后增寄存器 （如用于表读和表写的 TBLPTR）。 后减寄存器 （如用于表读和表写的 TBLPTR）。 预增寄存器 （如用于表读和表写的 TBLPTR）。
n	相对跳转指令的相对地址 （二进制补码），或 Call/Branch 和 Return 指令的直接地址。
PC	程序计数器。
PCL	程序计数器低字节。
PCH	程序计数器高字节。
PCLATH	程序计数器高字节锁存器。
PCLATU	程序计数器最高字节锁存器。
$\overline{\text{PD}}$	掉电位。
PRODH	乘积的高字节。
PRODL	乘积的低字节。
s	快速调用 / 返回模式选择位。 s = 0：不对影子寄存器进行更新，也不用影子寄存器的内容更新其他寄存器 s = 1：将寄存器的值存入影子寄存器或把影子寄存器中的值载入寄存器 （快速模式）
TBLPTR	21 位表指针 （指向程序存储器地址）。
TABLAT	8 位表锁存器。
TO	超时溢出位。
TOS	栈顶。
u	未使用或未改变。
WDT	看门狗定时器。
WREG	工作寄存器 （累加器）。
x	忽略 （0 或 1）。汇编器将产生 x = 0 的代码。为了与所有的 Microchip 软件工具兼容，建议使用这种格式。
z_s	对寄存器 （源）进行间接寻址的 7 位偏移量。
z_d	对寄存器 （目标）进行间接寻址的 7 位偏移量。
{ }	可选参数。
[text]	表示变址地址。
（text）	text 的内容。
[expr]<n>	表示由指针 expr 指定的寄存器中的位 n。
→	赋值。
< >	寄存器位域。
∈	表示属于某个集合。
斜体文字	用户定义项 （字体为 Courier）。

图 24-1： 指令的通用格式

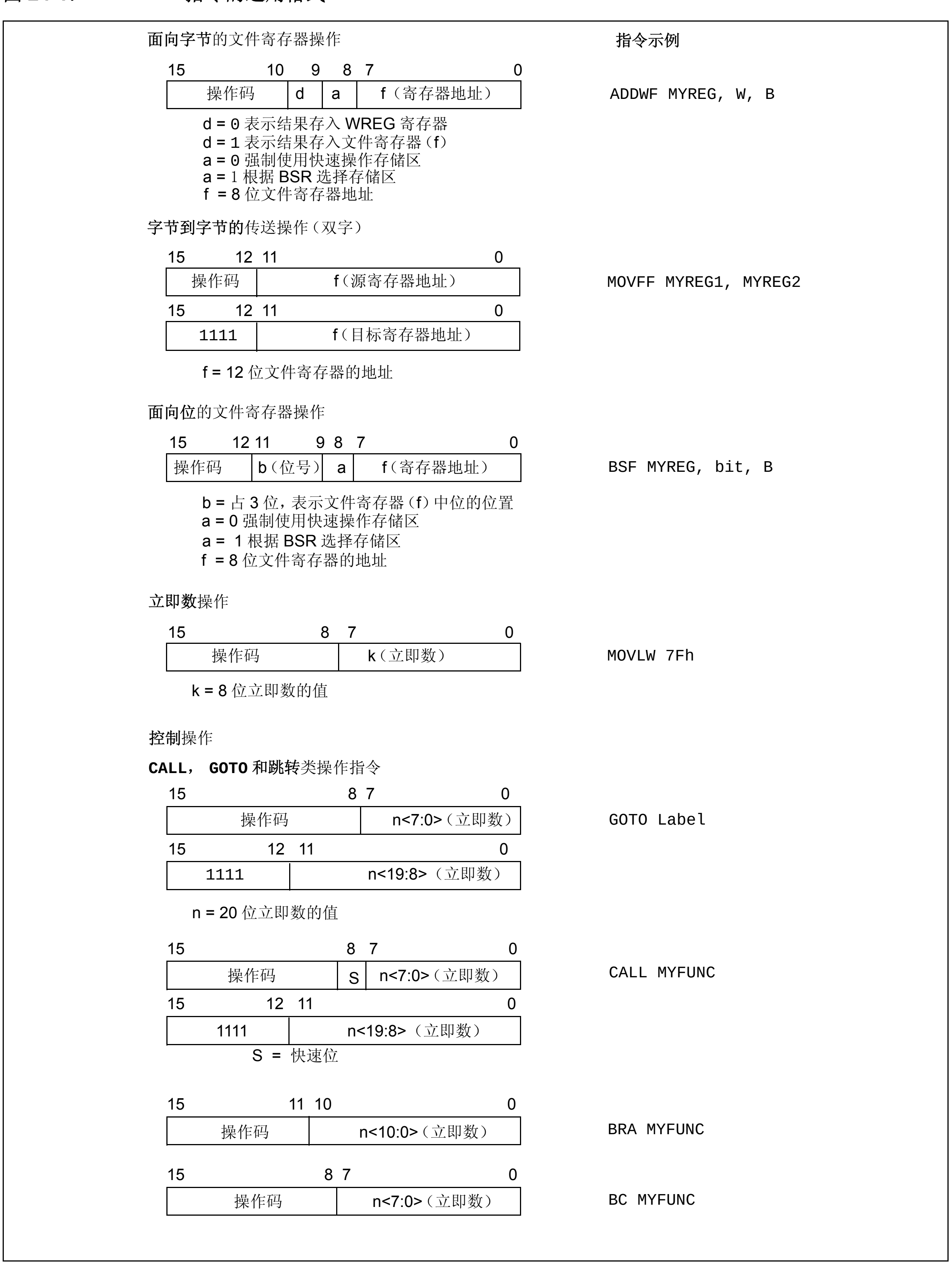


表 24-2： PIC18FXXXX 指令集

助记符，操作数		说明	周期	16 位指令字 MSb			LSb	受影响的状态位	注
面向字节的操作类指令									
ADDWF	f, d, a	WREG 与 f 相加	1	0010	01da	ffff	ffff	C, DC, Z, OV, N	1, 2
ADDWFC	f, d, a	WREG 与 f 带进位相加	1	0010	00da	ffff	ffff	C, DC, Z, OV, N	1, 2
ANDWF	f, d, a	WREG 和 F 做与运算	1	0001	01da	ffff	ffff	Z, N	1,2
CLRF	f, a	f 清零	1	0110	101a	ffff	ffff	Z	2
COMF	f, d, a	f 取补	1	0001	11da	ffff	ffff	Z, N	1, 2
CPFSEQ	f, a	将 f 与 WREG 做比较，相等则跳过	1 （2 或 3）	0110	001a	ffff	ffff	无	4
CPFSGT	f, a	将 f 与 WREG 做比较，大于则跳过	1 （2 或 3）	0110	010a	ffff	ffff	无	4
CPFSLT	f, a	将 f 与 WREG 做比较，小于则跳过	1 （2 或 3）	0110	000a	ffff	ffff	无	1, 2
DECF	f, d, a	f 减 1	1	0000	01da	ffff	ffff	C, DC, Z, OV, N	1, 2, 3, 4
DECFSZ	f, d, a	f 减 1，为 0 则跳过	1 （2 或 3）	0010	11da	ffff	ffff	无	1, 2, 3, 4
DCFSNZ	f, d, a	f 减 1，非 0 则跳过	1 （2 或 3）	0100	11da	ffff	ffff	无	1, 2
INCF	f, d, a	f 加 1	1	0010	10da	ffff	ffff	C, DC, Z, OV, N	1, 2, 3, 4
INCFSZ	f, d, a	f 加 1，为 0 则跳过	1 （2 或 3）	0011	11da	ffff	ffff	无	4
INFSNZ	f, d, a	f 加 1，非 0 则跳过	1 （2 或 3）	0100	10da	ffff	ffff	无	1, 2
IORWF	f, d, a	WREG 和 f 做或运算	1	0001	00da	ffff	ffff	Z, N	1, 2
MOVF	f, d, a	移动 f	1	0101	00da	ffff	ffff	Z, N	1
MOVFF	f_s, f_d	f_s （源）地址装入 （第一个字） f_d （目标）地址装入（第二个字）	2	1100 1111	ffff ffff	ffff ffff	ffff ffff	无	
MOVWF	f, a	将 WREG 移入 f	1	0110	111a	ffff	ffff	无	
MULWF	f, a	WREG 乘以 f	1	0000	001a	ffff	ffff	无	1, 2
NEGF	f, a	将 f 取负	1	0110	110a	ffff	ffff	C, DC, Z, OV, N	
RLCF	f, d, a	对 f 执行带进位的循环左移	1	0011	01da	ffff	ffff	C, Z, N	1, 2
RLNCF	f, d, a	f 循环左移 （不带进位）	1	0100	01da	ffff	ffff	Z, N	
RRCF	f, d, a	对 f 执行带进位的循环右移	1	0011	00da	ffff	ffff	C, Z, N	
RRNCF	f, d, a	f 循环右移 （不带进位）	1	0100	00da	ffff	ffff	Z, N	
SETF	f, a	将 f 置为全 1	1	0110	100a	ffff	ffff	无	1, 2
SUBFWB	f, d, a	WREG 减去 f （带借位）	1	0101	01da	ffff	ffff	C, DC, Z, OV, N	
SUBWF	f, d, a	f 减去 WREG	1	0101	11da	ffff	ffff	C, DC, Z, OV, N	1, 2
SUBWFB	f, d, a	f 减去 WREG （带借位）	1	0101	10da	ffff	ffff	C, DC, Z, OV, N	
SWAPF	f, d, a	将 f 中的两个半字节进行交换	1	0011	10da	ffff	ffff	无	4
TSTFSZ	f, a	测试 f，为 0 则跳过	1 （2 或 3）	0110	011a	ffff	ffff	无	1, 2
XORWF	f, d, a	WREG 和 f 异或运算	1	0001	10da	ffff	ffff	Z, N	

注 1： Port 寄存器的值随端口状态的变化而不断修改 （例如， MOVF PORTB, 1, 0），修改时使用的值是引脚上的当前值。例如，如果将一引脚配置为输入，其对应数据锁存器中的值将为 1，但此时若有外部器件将该引脚驱动为低电平，则被写回数据锁存器的数据值将是 0。

2： 当对 TMR0 寄存器执行该指令 （并且 d = 1）时，如果已为其分配了预分频器，则将该预分频器清零。

3： 如果程序计数器 （PC）被修改或者条件检测为 true，则该指令需要两个周期。第二个周期执行为一条 NOP 指令。

4： 某些指令是双字指令。除非指令的第一个字获取这 16 位中包含的信息，否则第二个字将作为 NOP 指令执行。这将确保所有程序存储器单元内存储的都是合法的指令。

5： 如果开始对内部存储器进行表写操作，则写操作将持续到终止为止。

表 **24-2：PIC18FXXXX** 指令集 （续）

助记符，操作数		说明	周期	16 位指令字 MSb			LSb	受影响的状态位	注
面向位的操作类指令									
BCF	f, b, a	将 f 中的某位清零	1	1001	bbba	ffff	ffff	无	1, 2
BSF	f, b, a	将 f 中的某位置 1	1	1000	bbba	ffff	ffff	无	1, 2
BTFSC	f, b, a	测试 f 中的某位，为 0 则跳过	1 （2 或 3）	1011	bbba	ffff	ffff	无	3, 4
BTFSS	f, b, a	测试 f 中的某位，为 1 则跳过	1 （2 或 3）	1010	bbba	ffff	ffff	无	3, 4
BTG	f, d, a	将 f 中的某位取反	1	0111	bbba	ffff	ffff	无	1, 2
控制类指令									
BC	n	进位则跳转	1 (2)	1110	0010	nnnn	nnnn	无	
BN	n	为负则跳转	1 (2)	1110	0110	nnnn	nnnn	无	
BNC	n	无进位则跳转	1 (2)	1110	0011	nnnn	nnnn	无	
BNN	n	不为负则跳转	1 (2)	1110	0111	nnnn	nnnn	无	
BNOV	n	不溢出则跳转	1 (2)	1110	0101	nnnn	nnnn	无	
BNZ	n	不为零则跳转	1 (2)	1110	0001	nnnn	nnnn	无	
BOV	n	溢出则跳转	1 (2)	1110	0100	nnnn	nnnn	无	
BRA	n	无条件跳转	2	1101	0nnn	nnnn	nnnn	无	
BZ	n	为零则跳转	1 (2)	1110	0000	nnnn	nnnn	无	
CALL	n, s	调用子程序 （第一个字）	2	1110	110s	kkkk	kkkk	无	
		（第二个字）		1111	kkkk	kkkk	kkkk		
CLRWDT	—	看门狗定时器清零	1	0000	0000	0000	0100	$\overline{TO}$，$\overline{PD}$	
DAW	—	对 WREG 进行十进制调整	1	0000	0000	0000	0111	C	
GOTO	n	跳转到地址 （第一个字）	2	1110	1111	kkkk	kkkk	无	
		（第二个字）		1111	kkkk	kkkk	kkkk		
NOP	—	空操作	1	0000	0000	0000	0000	无	
NOP	—	空操作	1	1111	xxxx	xxxx	xxxx	无	4
POP	—	从返回堆栈栈顶 （TOS）出栈	1	0000	0000	0000	0110	无	
PUSH	—	从返回堆栈栈顶 （TOS）进栈	1	0000	0000	0000	0101	无	
RCALL	n	相对调用	2	1101	1nnn	nnnn	nnnn	无	
RESET		用软件使器件复位	1	0000	0000	1111	1111	全部	
RETFIE	s	中断返回使能	2	0000	0000	0001	000s	GIE/GIEH, PEIE/GIEL	
RETLW	k	返回时将立即数送入 WREG	2	0000	1100	kkkk	kkkk	无	
RETURN	s	从子程序返回	2	0000	0000	0001	001s	无	
SLEEP	—	进入待机模式	1	0000	0000	0000	0011	$\overline{TO}$，$\overline{PD}$	

注 **1：** Port 寄存器的值随端口状态的变化而不断修改 （例如， `MOVF PORTB, 1, 0`），修改时使用的值是引脚上的当前值。例如，如果将一引脚配置为输入，其对应数据锁存器中的值将为 `1`，但此时若有外部器件将该引脚驱动为低电平，则被写回数据锁存器的数据值将是 `0`。

2： 当对 TMR0 寄存器执行该指令 （并且 d = 1）时，如果已为其分配了预分频器，则将该预分频器清零。

3： 如果程序计数器 （PC）被修改或者条件检测为 true，则该指令需要两个周期。第二个周期执行为一条 `NOP` 指令。

4： 某些指令是双字指令。除非指令的第一个字获取这 16 位中包含的信息，否则第二个字将作为 `NOP` 指令执行。这将确保所有程序存储器单元内存储的都是合法的指令。

5： 如果开始对内部存储器进行表写操作，则写操作将持续到终止为止。

助记符，操作数		说明	周期	16 位指令字 MSb			LSb	受影响的状态位	注
立即数操作类指令									
ADDLW	k	WREG 与立即数相加	1	0000	1111	kkkk	kkkk	C, DC, Z, OV, N	
ANDLW	k	WREG 和立即数进行与运算	1	0000	1011	kkkk	kkkk	Z, N	
IORLW	k	WREG 和立即数进行或运算	1	0000	1001	kkkk	kkkk	Z, N	
LFSR	f, k	移动立即数 （12 位）（第二个字）到 FSR （f） （第一个字）	2	1110 1111	1110 0000	00ff kkkk	kkkk kkkk	无	
MOVLB	k	将立即数移入 BSR<3:0>	1	0000	0001	0000	kkkk	无	
MOVLW	k	将立即数移入 WREG	1	0000	1110	kkkk	kkkk	无	
MULLW	k	WREG 和立即数相乘	1	0000	1101	kkkk	kkkk	无	
RETLW	k	返回时将立即数送入 WREG	2	0000	1100	kkkk	kkkk	无	
SUBLW	k	立即数减去 WREG	1	0000	1000	kkkk	kkkk	C, DC, Z, OV, N	
XORLW	k	WREG 和立即数进行异或运算	1	0000	1010	kkkk	kkkk	Z, N	
数据存储器 ↔ 程序存储器操作									
TBLRD*		表读	2	0000	0000	0000	1000	无	
TBLRD*+		后增表读		0000	0000	0000	1001	无	
TBLRD*-		后减表读		0000	0000	0000	1010	无	
TBLRD+*		预增表读		0000	0000	0000	1011	无	
TBLWT*		表写	2	0000	0000	0000	1100	无	5
TBLWT*+		后增表写		0000	0000	0000	1101	无	5
TBLWT*-		后减表写		0000	0000	0000	1110	无	5
TBLWT+*		预增表写		0000	0000	0000	1111	无	5

注 **1：** Port 寄存器的值随端口状态的变化而不断修改 （例如， `MOVF PORTB, 1, 0`），修改时使用的值是引脚上的当前值。例如，如果将一引脚配置为输入，其对应数据锁存器中的值将为 `1`，但此时若有外部器件将该引脚驱动为低电平，则被写回数据锁存器的数据值将是 `0`。

2： 当对 TMR0 寄存器执行该指令 （并且 d = 1）时，如果已为其分配了预分频器，则将该预分频器清零。

3： 如果程序计数器 （PC）被修改或者条件检测为 true，则该指令需要两个周期。第二个周期执行为一条 `NOP` 指令。

4： 某些指令是双字指令。除非指令的第一个字获取这 16 位中包含的信息，否则第二个字将作为 `NOP` 指令执行。这将确保所有程序存储器单元内存储的都是合法的指令。

5： 如果开始对内部存储器进行表写操作，则写操作将持续到终止为止。

24.1.1 标准指令集

ADDLW　　W 与立即数相加

语法：　ADDLW　k

操作数：　$0 \leq k \leq 255$

操作：　（W）+ k→ W

受影响的状态位：　N、OV、C、DC 和 Z

机器码：

0000	1111	kkkk	kkkk

说明：　将 W 寄存器的内容与 8 位立即数 “k” 相加，结果保存在 W 寄存器。

指令字数：　1

指令周期数：　1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读立即数 “k”	处理数据	写入 W

示例：　ADDLW　15h

执行指令前
W　=　10h
执行指令后
W　=　25h

ADDWF　　W 与 f 寄存器相加

语法：　ADDWF　f {,d {,a}}

操作数：　$0 \leq f \leq 255$
d ∈ [0,1]
a ∈ [0,1]

操作：　（W）+ （f）→ dest

受影响的状态位：　N、OV、C、DC 和 Z

机器码：

0010	01da	ffff	ffff

说明：　将 W 的内容与 f 寄存器的内容相加。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f（默认）。
如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。
如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 24.2.3 节 “立即数变址寻址模式中面向字节和面向位的指令”**。

指令字数：　1

指令周期数：　1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 “f”	处理数据	写入目标寄存器

示例：　ADDWF　REG,　0, 0

执行指令前
W　=　17h
REG　=　0C2h
执行指令后
W　=　0D9h
REG　=　0C2h

注：	所有的 PIC18 指令都可能在其指令助记符之前使用可选的标号参数，用于符号寻址。如果使用了标号，那么指令语法将变为：{label} 指令参数。

ADDWFC W 与 f 带进位相加

语法：	ADDWFC f {,d {,a}}
操作数：	0 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]
操作：	（W） + （f） + （C） → dest
受影响的状态位：	N、 OV、C、DC 和 Z
机器码：	0010 00da ffff ffff
说明：	将 W 的内容、进位标志位与寄存器 f 的内容相加。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存储在寄存器 f 中。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。 如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 24.2.3 节 “立即数变址寻址模式中面向字节和面向位的指令”**。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 f	处理数据	写入 目标寄存器

示例： ADDWFC REG, 0, 1

执行指令前
- 进位标志位 = 1
- REG = 02h
- W = 4Dh

执行指令后
- 进位标志位 = 0
- REG = 02h
- W = 50h

ANDLW 立即数和 W 寄存器作逻辑与运算

语法：	ANDLW k
操作数：	0 ≤ k ≤ 255
操作：	（W） .AND.k → W
受影响的状态位：	N、 Z
机器码：	0000 1011 kkkk kkkk
说明：	将 W 的内容与 8 位立即数 k 进行逻辑与运算。结果保存在 W 寄存器中。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读立即数 k	处理数据	写入 W

示例： ANDLW 05Fh

执行指令前
- W = A3h

执行指令后
- W = 03h

ANDWF　将 W 和 f 作逻辑与运算

语法：	ANDWF　f {,d {,a}}
操作数：	0 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]
操作：	（W）.AND.（f）→ dest
受影响的状态位：	N、Z
机器码：	0001　01da　ffff　ffff
说明：	将 W 的内容与寄存器 f 的内容进行逻辑与运算。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f（默认）。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。 如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95（5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 24.2.3 节"立即数变址寻址模式中面向字节和面向位的指令"**。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 f	处理数据	写入 目标寄存器

示例：　ANDWF　REG,　0, 0

执行指令前
W = 17h
REG = C2h
执行指令后
W = 02h
REG = C2h

BC　进位则跳转

语法：	BC　n
操作数：	-128 ≤ n ≤ 127
操作：	如果进位标志位为 1 （PC）+ 2 + 2n → PC
受影响的状态位：	无
机器码：	1110　0010　nnnn　nnnn
说明：	如果进位标志位为 1，程序将跳转。二进制补码"2n"与 PC 相加。由于 PC 将递增以便取出下一条指令，所以新地址将为 PC + 2 + 2n。进位标志位为 1 时，该指令为一条双周期指令。
指令字数：	1
指令周期数：	1（2）

Q 周期操作：
如果跳转：

Q1	Q2	Q3	Q4
译码	读立即数 n	处理数据	写入 PC
空操作	空操作	空操作	空操作

如果不跳转：

Q1	Q2	Q3	Q4
译码	读立即数 n	处理数据	空操作

示例：　HERE　BC　5

执行指令前
PC = 地址（HERE）
执行指令后
如果进位标志位 = 1；
PC = 地址（HERE + 12）
如果进位标志位 = 0；
PC = 地址（HERE + 2）

BCF 将 f 寄存器中的某位清零

语法：	BCF f, b {,a}
操作数：	0 ≤ f ≤ 255 0 ≤ b ≤ 7 a ∈ [0,1]
操作：	0 → f<b>
受影响的状态位：	无
机器码：	1001 bbba ffff ffff
说明：	将寄存器 f 中的位 b 清零。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。 如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95（5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 24.2.3 节“立即数变址寻址模式中面向字节和面向位的指令”**。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写寄存器 f

示例： `BCF FLAG_REG, 7, 0`

执行指令前
FLAG_REG = C7h
执行指令后
FLAG_REG = 47h

BN 为负则跳转

语法：	BN n
操作数：	-128 ≤ n ≤ 127
操作：	如果负标志位为 1 （PC）+ 2 + 2n → PC
受影响的状态位：	无
机器码：	1110 0110 nnnn nnnn
说明：	如果负标志位为 1，程序将跳转。 二进制补码 “2n” 与 PC 相加。由于 PC 将递增以便取出下一条指令，所以新地址将为 PC + 2 + 2n。负标志位为 1 时，该指令为一条双周期指令。
指令字数：	1
指令周期数：	1（2）

Q 周期操作：
如果跳转：

Q1	Q2	Q3	Q4
译码	读立即数 n	处理数据	写入 PC
空操作	空操作	空操作	空操作

如果不跳转：

Q1	Q2	Q3	Q4
译码	读立即数 n	处理数据	空操作

示例： `HERE BN Jump`

执行指令前
PC = 地址 （HERE）
执行指令后
如果负标志位 = 1；
PC = 地址（Jump）
如果负标志位 = 0；
PC = 地址（HERE + 2）

 初稿

BNC 无进位则跳转

语法：	BNC n
操作数：	-128 ≤ n ≤ 127
操作：	如果进位标志位为 0 （PC）+ 2 + 2n → PC
受影响的状态位：	无
机器码：	1110 0011 nnnn nnnn
说明：	如果进位标志位为 0，程序将跳转。二进制补码 “2n” 与 PC 相加。由于 PC 将递增以便取出下一条指令，所以新地址将为 PC + 2 + 2n。进位标志位为 0 时，该指令为一条双周期指令。
指令字数：	1
指令周期数：	1（2）

Q 周期操作：

如果跳转：

Q1	Q2	Q3	Q4
译码	读立即数 n	处理数据	写入 PC
空操作	空操作	空操作	空操作

如果不跳转：

Q1	Q2	Q3	Q4
译码	读立即数 n	处理数据	空操作

示例： HERE BNC Jump

执行指令前
 PC = 地址 （HERE）
执行指令后
 如果进位标志位 = 0；
 PC = 地址 （Jump）
 如果进位标志位 = 1；
 PC = 地址 （HERE +2）

BNN 不为负则跳转

语法：	BNN n
操作数：	-128 ≤ n ≤ 127
操作：	如果负标志位为 0 （PC）+ 2 + 2n → PC
受影响的状态位：	无
机器码：	1110 0111 nnnn nnnn
说明：	如果负标志位为 0，程序将跳转。二进制补码 “2n” 与 PC 相加。由于 PC 将递增以便取出下一条指令，所以新地址将为 PC + 2 + 2n。负标志位为 0 时，该指令为一条双周期指令。
指令字数：	1
指令周期数：	1（2）

Q 周期操作：

如果跳转：

Q1	Q2	Q3	Q4
译码	读立即数 n	处理数据	写入 PC
空操作	空操作	空操作	空操作

如果不跳转：

Q1	Q2	Q3	Q4
译码	读立即数 n	处理数据	空操作

示例： HERE BNN Jump

执行指令前
 PC = 地址 （HERE）
执行指令后
 如果负标志位 = 0；
 PC = 地址 （Jump）
 如果负标志位 = 1；
 PC = 地址 （HERE +2）

BNOV 不溢出则跳转

语法：	BNOV n
操作数：	-128 ≤ n ≤ 127
操作：	如果溢出标志位为 0 （PC）+ 2 + 2n → PC
受影响的状态位：	无

机器码：

1110	0101	nnnn	nnnn

说明：如果溢出标志位为 0，程序将跳转。二进制补码 “2n” 与 PC 相加。由于 PC 将递增以便取出下一条指令，所以新地址将为 PC + 2 + 2n。溢出标志位为 0 时，该指令为一条双周期指令。

指令字数：1

指令周期数：1（2）

Q 周期操作：
如果跳转：

Q1	Q2	Q3	Q4
译码	读立即数 n	处理数据	写入 PC
空操作	空操作	空操作	空操作

如果不跳转：

Q1	Q2	Q3	Q4
译码	读立即数 n	处理数据	空操作

示例： HERE BNOV Jump

```
执行指令前
    PC             = 地址 （HERE）
执行指令后
    如果溢出标志位 = 0；
        PC         = 地址 （Jump）
    如果溢出标志位 = 1；
        PC         = 地址 （HERE + 2）
```

BNZ 不为零则跳转

语法：	BNZ n
操作数：	-128 ≤ n ≤ 127
操作：	如果全零标志位为 0 （PC）+ 2 + 2n → PC
受影响的状态位：	无

机器码：

1110	0001	nnnn	nnnn

说明：如果全零标志位为 0，程序将跳转。二进制补码 “2n” 与 PC 相加。由于 PC 将递增以便取出下一条指令，所以新地址将为 PC + 2 + 2n。全零标志位为 0 时，该指令为一条双周期指令。

指令字数：1

指令周期数：1（2）

Q 周期操作：
如果跳转：

Q1	Q2	Q3	Q4
译码	读立即数 n	处理数据	写入 PC
空操作	空操作	空操作	空操作

如果不跳转：

Q1	Q2	Q3	Q4
译码	读立即数 n	处理数据	空操作

示例： HERE BNZ Jump

```
执行指令前
    PC             = 地址 （HERE）
执行指令后
    如果全零标志位 = 0；
        PC         = 地址 （Jump）
    如果全零标志位 = 1；
        PC         = 地址 （HERE + 2）
```

BRA 无条件跳转

语法：	BRA n
操作数：	-1024 ≤ n ≤ 1023
操作：	（PC）+ 2 + 2n → PC
受影响的状态位：	无
机器码：	`1101` `0nnn` `nnnn` `nnnn`
说明：	二进制补码 “2n” 与 PC 相加。因为 PC 要先递增才能取下一条指令，所以新地址将为 PC + 2 + 2n。该指令为一条双周期指令。
指令字数：	1
指令周期数：	2

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读立即数 n	处理数据	写入 PC
空操作	空操作	空操作	空操作

示例： `HERE BRA Jump`

执行指令前
PC = 地址 （`HERE`）
执行指令后
PC = 地址 （`Jump`）

BSF 将 f 寄存器中的某位置 1

语法：	BSF f, b {,a}
操作数：	0 ≤ f ≤ 255 0 ≤ b ≤ 7 a ∈ [0,1]
操作：	1 → f<b>
受影响的状态位：	无
机器码：	`1000` `bbba` `ffff` `ffff`
说明：	将寄存器 f 的位 b 置 1。 如果 a 为 `0`，选择快速操作存储区。如果 a 为 `1`，使用 BSR 选择 GPR 存储区（默认）。 如果 a 为 `0` 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 24.2.3 节“立即数变址寻址模式中面向字节和面向位的指令”**。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写寄存器 f

示例： `BSF FLAG_REG, 7, 1`

执行指令前
FLAG_REG = 0Ah
执行指令后
FLAG_REG = 8Ah

BTFSC 测试寄存器中的位，为 0 则跳过

语法：	BTFSC f, b {,a}
操作数：	0 ≤ f ≤ 255 0 ≤ b ≤ 7 a ∈ [0,1]
操作：	如果 （f<b>）= 0 则跳过
受影响的状态位：	无
机器码：	1011 bbba ffff ffff
说明：	如果寄存器 f 的位 b 为 0，则跳过下一条指令。即在 b 位为 0 时，丢弃下一条指令（执行当前指令期间取的指令）而执行一条 NOP 指令，使该指令变成双周期指令。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区 （默认）。 如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 24.2.3 节 “立即数变址寻址模式中面向字节和面向位的指令”**。
指令字数：	1
指令周期数：	1 （2） 注：如果跳过的指令后面跟有 2 字指令，则执行 BTFSC 需要 3 个周期。

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 f	处理 数据	空操作

如果跳过：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作

如果跳过的指令后面跟有 2 字指令：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作
空操作	空操作	空操作	空操作

示例：

```
HERE    BTFSC   FLAG, 1, 0
FALSE   :
TRUE    :
```

执行指令前
PC = 地址 （HERE）

执行指令后
如果 FLAG<1> = 0；
PC = 地址 （TRUE）
如果 FLAG<1> = 1；
PC = 地址 （FALSE）

BTFSS 测试寄存器中的位，为 1 则跳过

语法：	BTFSS f, b {,a}
操作数：	0 ≤ f ≤ 255 0 ≤ b < 7 a ∈ [0,1]
操作：	如果 （f<b>） = 1 则跳过
受影响的状态位：	无
机器码：	1010 bbba ffff ffff
说明：	如果寄存器 f 的位 b 为 1，则跳过下一条指令。即在 b 位为 1 时，丢弃下一条指令（执行当前指令期间取的指令）而执行一条 NOP 指令，使该指令变成双周期指令。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区 （默认）。 如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 24.2.3 节 “立即数变址寻址模式中面向字节和面向位的指令”**。
指令字数：	1
指令周期数：	1 （2） 注：如果跳过的指令后面跟有 2 字指令，则执行 BTFSS 需要 3 个周期。

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 f	处理 数据	空操作

如果跳过：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作

如果跳过的指令后面跟有 2 字指令：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作
空操作	空操作	空操作	空操作

示例：

```
HERE    BTFSS   FLAG, 1, 0
FALSE   :
TRUE    :
```

执行指令前
PC = 地址 （HERE）

执行指令后
如果 FLAG<1> = 0；
PC = 地址 （FALSE）
如果 FLAG<1> = 1；
PC = 地址 （TRUE）

BTG　　将 f 中的位取反

语法：　BTG f, b {,a}

操作数：　0 ≤ f ≤ 255
0 ≤ b < 7
a ∈ [0,1]

操作：　（$\overline{\text{f<b>}}$）→ f<b>

受影响的状态位：　无

机器码：

0111	bbba	ffff	ffff

说明：　将数据存储单元 f 中的位 b 取反。
如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。
如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 24.2.3 节 “立即数变址寻址模式中面向字节和面向位的指令”**。

指令字数：　1

指令周期数：　1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 f	处理数据	写 寄存器 f

<u>示例：</u>　`BTG PORTC, 4, 0`

执行指令前：
　PORTC = `0111 0101` [75h]
执行指令后：
　PORTC = `0110 0101` [65h]

BOV　　溢出则跳转

语法：　BOV n

操作数：　-128 ≤ n ≤ 127

操作：　如果溢出标志位为 1
（PC）+ 2 + 2n → PC

受影响的状态位：　无

机器码：

1110	0100	nnnn	nnnn

说明：　如果溢出标志位为 1，程序将跳转。
二进制补码 “2n” 与 PC 相加。由于 PC 将递增以便取出下一条指令，所以新地址将为 PC + 2 + 2n。溢出标志位为 1 时，该指令为一条双周期指令。

指令字数：　1

指令周期数：　1 （2）

Q 周期操作：
如果跳转：

Q1	Q2	Q3	Q4
译码	读立即数 n	处理数据	写入 PC
空操作	空操作	空操作	空操作

如果不跳转：

Q1	Q2	Q3	Q4
译码	读立即数 n	处理数据	空操作

<u>示例：</u>　`HERE BOV Jump`

执行指令前
　PC = 地址 （`HERE`）
执行指令后
　如果溢出标志位 = 1；
　　PC = 地址 （`Jump`）
　如果溢出标志位 = 0；
　　PC = 地址 （`HERE + 2`）

BZ 为零则跳转

语法：	BZ n
操作数：	-128 ≤ n ≤ 127
操作：	如果全零标志位为 1 （PC）+ 2 + 2n → PC
受影响的状态位：	无
机器码：	1110 0000 nnnn nnnn
说明：	如果全零标志位为 1，程序将跳转。二进制补码 “2n” 与 PC 相加。由于 PC 将递增以便取出下一条指令，所以新地址将为 PC + 2 + 2n。全零标志位为 1 时，该指令为一条双周期指令。
指令字数：	1
指令周期数：	1（2）

Q 周期操作：

如果跳转：

Q1	Q2	Q3	Q4
译码	读立即数 n	处理数据	写入 PC
空操作	空操作	空操作	空操作

如果不跳转：

Q1	Q2	Q3	Q4
译码	读立即数 “n”	处理数据	空操作

示例： HERE BZ Jump

执行指令前
  PC = 地址 （HERE）
执行指令后
  如果全零标志位 = 1；
  PC = 地址 （Jump）
  如果全零标志位 = 0；
  PC = 地址 （HERE + 2）

CALL 调用子程序

语法：	CALL k {,s}
操作数：	0 ≤ k ≤ 1048575 s ∈ [0,1]
操作：	（PC）+ 4 → TOS, k → PC<20:1>, 如果 s = 1 （W）→ WS, （Status 寄存器）→ STATUSS, （BSR）→ BSRS
受影响的状态位：	无

机器码：

第一个字 （k<7:0>）	1110	110s	$k_7$kkk	kkkk$_0$
第二个字 （k<19:8>）	1111	k_{19}kkk	kkkk	kkkk$_8$

说明： 可在整个 2 MB 的存储器范围内进行子程序调用。首先，将返回地址 （PC+ 4）压入返回堆栈。如果 s = 1，还会将 W、 Status 和 BSR 寄存器的内容存入它们各自的影子寄存器 WS、 STATUSS 和 BSRS。如果 s = 0，将不会进行任何更新 （默认）。然后，将 20 位的值 k 装入 PC<20:1>。 CALL 是一条双周期指令。

指令字数： 2

指令周期数： 2

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读立即数 k<7:0>	将 PC 压入堆栈	读立即数 k<19:8>，写入 PC
空操作	空操作	空操作	空操作

示例： HERE CALL THERE,1

执行指令前
  PC = 地址 （HERE）
执行指令后
  PC = 地址 （THERE）
  TOS = 地址 （HERE + 4）
  WS = W 寄存器
  BSRS = BSR 寄存器
  STATUSS = Status 寄存器

CLRF　将 f 清零

语法：	CLRF　f {,a}
操作数：	0 ≤ f ≤ 255 a ∈ [0,1]
操作：	000h → f 1 → Z
受影响的状态位：	Z
机器码：	0110 101a ffff ffff
说明：	清零指定寄存器的内容。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。 如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95（5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 24.2.3 节“立即数变址寻址模式中面向字节和面向位的指令”**。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 f	处理数据	写 寄存器 f

示例：　CLRF　FLAG_REG,1

执行指令前
FLAG_REG = 5Ah
执行指令后
FLAG_REG = 00h

CLRWDT　将看门狗定时器清零

语法：	CLRWDT
操作数：	无
操作：	000h → WDT， 000h → WDT 后分频器， 1 → $\overline{TO}$， 1 → $\overline{PD}$
受影响的状态位：	$\overline{TO}$，$\overline{PD}$
机器码：	0000 0000 0000 0100
说明：	CLRWDT 指令复位看门狗定时器及其后分频器。状态位 $\overline{TO}$ 和 $\overline{PD}$ 置 1。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	空 操作	处理数据	空操作

示例：　CLRWDT

执行指令前
WDT 计数器 = ?
执行指令后
WDT 计数器 = 00h
WDT 后分频器 = 0
$\overline{TO}$ = 1
$\overline{PD}$ = 1

COMF **将 f 取补**

语法：	COMF f {,d {,a}}
操作数：	0 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]
操作：	$(\overline{f})$ → dest
受影响的状态位：	N、Z
机器码：	0001 11da ffff ffff
说明：	将寄存器 f 的内容取补。如果 d 为 1，结果存储在 W 中。如果 d 为 0，结果存回寄存器 f（默认）。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。 如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95（5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 24.2.3 节“立即数变址寻址模式中面向字节和面向位的指令”**。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 f	处理数据	写入 目标寄存器

示例： `COMF REG, 0, 0`

执行指令前
REG = 13h
执行指令后
REG = 13h
W = ECh

CPFSEQ **比较 f 和 W，如果 f = W 则跳过**

语法：	CPFSEQ f {,a}
操作数：	0 ≤ f ≤ 255 a ∈ [0,1]
操作：	（f） – （W）， 如果（f） = （W）则跳过 （无符号比较）
受影响的状态位：	无
机器码：	0110 001a ffff ffff
说明：	通过执行无符号的减法，将数据存储单元 f 的内容与 W 的内容做比较。 如果 f = W，则所取的指令被丢弃并执行一条 NOP 指令，使这个指令成为双周期指令。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。 如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95（5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 24.2.3 节“立即数变址寻址模式中面向字节和面向位的指令”**。
指令字数：	1
指令周期数：	1（2） 注：如果跳过的指令后面跟有 2 字指令，则执行 CPFSEQ 需要 3 个周期。

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 f	处理数据	空操作

如果跳过：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作

如果跳过的指令后面跟有 2 字指令：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作
空操作	空操作	空操作	空操作

示例：

```
HERE     CPFSEQ REG, 0
NEQUAL   :
EQUAL    :
```

执行指令前
PC 地址 = HERE
W = ?
REG = ?
执行指令后
如果 REG = W；
PC = 地址（EQUAL）
如果 REG ≠ W；
PC = 地址（NEQUAL）

CPFSGT 比较 f 和 W，如果 f > W 则跳过

语法：	CPFSGT f {,a}
操作数：	0 ≤ f ≤ 255 a ∈ [0,1]
操作：	（f） – （W）， 如果 （f） > （W），则跳过 （无符号比较）
受影响的状态位：	无
机器码：	0110 010a ffff ffff
说明：	通过执行无符号的减法，将数据存储单元 f 的内容与 W 的内容做比较。 如果 f>W，则所取的指令被丢弃并执行一条 NOP 指令，使这个指令成为双周期指令。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。 如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 24.2.3 节“立即数变址寻址模式中面向字节和面向位的指令”**。
指令字数：	1
指令周期数：	1 （2） **注：**如果跳过的指令后面跟有 2 字指令，则执行 CPFSGT 需要 3 个周期。

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 f	处理数据	空 操作

如果跳过：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作

如果跳过的指令后面跟有 2 字指令：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作
空操作	空操作	空操作	空操作

示例：

```
HERE       CPFSGT REG, 0
NGREATER   :
GREATER    :
```

执行指令前
PC = 地址 （HERE）
W = ?

执行指令后
如果 REG > W；
PC = 地址 （GREATER）
如果 REG ≤ W；
PC = 地址 （NGREATER）

CPFSLT 比较 f 和 W，如果 f < W 则跳过

语法：	CPFSLT f {,a}
操作数：	0 ≤ f ≤ 255 a ∈ [0,1]
操作：	（f） – （W）， 如果 （f） < （W）则跳过 （无符号比较）
受影响的状态位：	无
机器码：	0110 000a ffff ffff
说明：	通过执行无符号的减法，将数据存储单元 f 的内容与 W 的内容做比较。 如果 f<W，则所取的指令被丢弃并执行一条 NOP 指令，使这个指令成为双周期指令。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。 如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 24.2.3 节“立即数变址寻址模式中面向字节和面向位的指令”**。
指令字数：	1
指令周期数：	1 （2） **注：**如果跳过的指令后面跟有 2 字指令，则执行 CPFSLT 需要 3 个周期。

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 f	处理数据	空 操作

如果跳过：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作

如果跳过的指令后面跟有 2 字指令：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作
空操作	空操作	空操作	空操作

示例：

```
HERE    CPFSLT REG, 1
NLESS   :
LESS    :
```

执行指令前
PC = 地址 （HERE）
W = ?

执行指令后
如果 REG < W；
PC = 地址 （LESS）
如果 REG ≥ W；
PC = 地址 （NLESS）

DAW 对 W 寄存器进行十进制调整

语法：	DAW
操作数：	无
操作：	如果 [W<3:0> >9] 或 [DC = 1]，则 （W<3:0>）+ 6 → W<3:0>； 否则 （W<3:0>）→ W<3:0>。 如果 [W<7:4> >9] 或 [C = 1]，则 （W<7:4>）+ 6 → W<7:4>， C = 1； 否则 （W<7:4>）→ W<7:4>。
受影响的状态位：	C
机器码：	0000 0000 0000 0111
说明：	DAW指令调整W内的8位值，即前两个压缩 BCD 格式的变量之和，并产生一个正确的压缩 BCD 格式结果。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 W 寄存器	处理数据	写 W 寄存器

例 1： DAW

执行指令前
W = A5h
C = 0
DC = 0
执行指令后
W = 05h
C = 1
DC = 0

例 2：

执行指令前
W = CEh
C = 0
DC = 0
执行指令后
W = 34h
C = 1
DC = 0

DECF f 减 1

语法：	DECF f {,d {,a}}
操作数：	0 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]
操作：	（f） – 1 → dest
受影响的状态位：	C、DC、N、OV 和 Z
机器码：	0000 01da ffff ffff
说明：	将寄存器 f 的内容减 1。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f （默认）。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。 如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 24.2.3 节“立即数变址寻址模式中面向字节和面向位的指令”**。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 f	处理数据	写入 目标寄存器

示例： DECF CNT, 1, 0

执行指令前
CNT = 01h
Z = 0
执行指令后
CNT = 00h
Z = 1

DECFSZ **f 减 1，为 0 则跳过**

语法：	DECFSZ f {,d {,a}}
操作数：	0 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]
操作：	（f）– 1 → dest， 结果为 0 时跳过
受影响的状态位：	无
机器码：	0010 11da ffff ffff
说明：	将寄存器 f 的内容减 1。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f（默认）。 如果结果为 0，则丢弃已经取指的指令而执行一条 NOP 指令，使该指令成为双周期指令。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。 如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95（5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 24.2.3 节 "立即数变址寻址模式中面向字节和面向位的指令"**。
指令字数：	1
指令周期数：	1（2） **注：**如果跳过的指令后面跟有 2 字指令，则执行 DECFSZ 需要 3 个周期。

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 f	处理数据	写入 目标寄存器

如果跳过：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作

如果跳过的指令后面跟有 2 字指令：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作
空操作	空操作	空操作	空操作

示例：

```
HERE      DECFSZ   CNT, 1, 1
          GOTO     LOOP
CONTINUE
```

执行指令前
PC = 地址（HERE）
执行指令后
CNT = CNT – 1
如果 CNT = 0；
PC = 地址（CONTINUE）
如果 CNT ≠ 0；
PC = 地址（HERE + 2）

DCFSNZ **f 减 1，非 0 则跳过**

语法：	DCFSNZ f {,d {,a}}
操作数：	0 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]
操作：	（f）– 1 → dest， 结果不为 0 时跳过
受影响的状态位：	无
机器码：	0100 11da ffff ffff
说明：	将寄存器 f 的内容减 1。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f（默认）。 如果结果不为 0，则丢弃已经取指的指令而执行一条 NOP 指令，使该指令成为双周期指令。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。 如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95（5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 24.2.3 节 "立即数变址寻址模式中面向字节和面向位的指令"**。
指令字数：	1
指令周期数：	1（2） **注：**如果跳过的指令后面跟有 2 字指令，则执行 DCFSNZ 需要 3 个周期。

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 f	处理数据	写入 目标寄存器

如果跳过：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作

如果跳过的指令后面跟有 2 字指令：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作
空操作	空操作	空操作	空操作

示例：

```
HERE    DCFSNZ  TEMP, 1, 0
ZERO    :
NZERO   :
```

执行指令前
TEMP = ?
执行指令后
TEMP = TEMP – 1，
如果 TEMP = 0；
PC = 地址（ZERO）
如果 TEMP ≠ 0；
PC = 地址（NZERO）

GOTO 无条件跳转

语法：	GOTO k
操作数：	0 ≤ k ≤ 1048575
操作：	k → PC<20:1>
受影响的状态位：	无

机器码：

第一个字 （k<7:0>）	1110	1111	$k_7$kkk	kkkk$_0$
第二个字 （k<19:8>）	1111	k_{19}kkk	kkkk	kkkk$_8$

说明： GOTO 指令允许无条件跳转到整个 2 MB 存储器范围中的任何位置。将 20 位值 k 装入 PC<20:1>。GOTO 始终为双周期指令。

指令字数： 2

指令周期数： 2

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读立即数 k<7:0>，	空操作	读立即数 k<19:8>，写入 PC
空操作	空操作	空操作	空操作

示例： `GOTO THERE`

执行指令后
PC = 地址 （THERE）

INCF f 加 1

语法：	INCF f {,d {,a}}
操作数：	0 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]
操作：	（f） + 1 → dest
受影响的状态位：	C、 DC、 N、 OV 和 Z

机器码：

0010	10da	ffff	ffff

说明： 将寄存器 f 的内容加 1。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f （默认）。
如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。
如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 24.2.3 节 "立即数变址寻址模式中面向字节和面向位的指令"**。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入目标寄存器

示例： `INCF CNT, 1, 0`

执行指令前
CNT = FFh
Z = 0
C = ?
DC = ?

执行指令后
CNT = 00h
Z = 1
C = 1
DC = 1

INCFSZ　　f 加 1，为 0 则跳过

语法：	INCFSZ　f {,d {,a}}
操作数：	0 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]
操作：	（f）+ 1 → dest， 如果结果为 0 则跳过
受影响的状态位：	无
机器码：	0011 11da ffff ffff
说明：	将寄存器 f 的内容加 1。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f（默认）。 如果结果为 0，则丢弃已经取指的指令而执行一条 NOP 指令，使该指令成为双周期指令。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。 如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95（5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 24.2.3 节“立即数变址寻址模式中面向字节和面向位的指令”**。
指令字数：	1
指令周期数：	1（2） **注：**如果跳过的指令后面跟有 2 字指令，则执行 INCFSZ 需要 3 个周期。

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入目标寄存器

如果跳过：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作

如果跳过的指令后面跟有 2 字指令：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作
空操作	空操作	空操作	空操作

示例：

```
HERE    INCFSZ   CNT, 1, 0
NZERO   :
ZERO    :
```

执行指令前
　PC　=　地址（HERE）

执行指令后
　CNT　=　CNT + 1
　如果 CNT =　0；
　PC　=　地址（ZERO）
　如果 CNT ≠　0；
　PC　=　地址（NZERO）

INFSNZ　　f 加 1，非 0 则跳过

语法：	INFSNZ　f {,d {,a}}
操作数：	0 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]
操作：	（f）+ 1 → dest， 如果结果不为 0 则跳过
受影响的状态位：	无
机器码：	0100 10da ffff ffff
说明：	将寄存器 f 的内容加 1。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f（默认）。 如果结果不为 0，则丢弃已经取指的指令而执行一条 NOP 指令，使该指令成为双周期指令。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。 如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95（5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 24.2.3 节“立即数变址寻址模式中面向字节和面向位的指令”**。
指令字数：	1
指令周期数：	1（2） **注：**如果跳过的指令后面跟有 2 字指令，则执行 INFSNZ 需要 3 个周期。

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入目标寄存器

如果跳过：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作

如果跳过的指令后面跟有 2 字指令：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作
空操作	空操作	空操作	空操作

示例：

```
HERE    INFSNZ   REG, 1, 0
ZERO
NZERO
```

执行指令前
　PC　=　地址（HERE）

执行指令后
　REG　=　REG + 1
　如果 REG≠　0；
　PC　=　地址（NZERO）
　如果 REG =　0；
　PC　=　地址（ZERO）

IORLW 将立即数与 W 作逻辑或运算

语法：	IORLW k
操作数：	0 ≤ k ≤ 255
操作：	（W）.OR. k → W
受影响的状态位：	N 和 Z
机器码：	0000	1001	kkkk	kkkk
说明：	将 W 的内容与 8 位立即数 k 进行逻辑或运算。结果保存在 W 寄存器中。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 立即数 k	处理数据	写 W

示例： `IORLW 35h`

执行指令前
W = 9Ah
执行指令后
W = BFh

IORWF 将 W 与 f 作逻辑或运算

语法：	IORWF f {,d {,a}}
操作数：	0 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]
操作：	（W）.OR.（f）→ dest
受影响的状态位：	N 和 Z
机器码：	0001	00da	ffff	ffff
说明：	将 W 的内容与寄存器 f 的内容进行逻辑或运算。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f（默认）。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。 如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95（5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 24.2.3 节“立即数变址寻址模式中面向字节和面向位的指令”**。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 f	处理数据	写入 目标寄存器

示例： `IORWF RESULT, 0, 1`

执行指令前
RESULT = 13h
W = 91h
执行指令后
RESULT = 13h
W = 93h

LFSR 载入 FSR

语法：	LFSR f, k
操作数：	0 ≤ f ≤ 2 0 ≤ k ≤ 4095
操作：	k → FSRf
受影响的状态位：	无

机器码：

1110	1110	00ff	k_{11}kkk
1111	0000	$k_7$kkk	kkkk

说明：	将 12 位立即数 k 载入 f 所指向的指针寄存器。
指令字数：	2
指令周期数：	2

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读立即数 k 的 MSB	处理数据	将立即数 k 的 MSB 写入 FSRfH
译码	读立即数 k 的 LSB	处理数据	将立即数 k 的 LSB 写入 FSRfL

示例： `LFSR 2, 3ABh`

执行指令后
FSR2H = 03h
FSR2L = ABh

MOVF 移动 f

语法：	MOVF f {,d {,a}}
操作数：	0 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]
操作：	f → dest
受影响的状态位：	N 和 Z

机器码：

0101	00da	ffff	ffff

说明： 根据 d 的状态，将寄存器 f 的内容移入目标单元。 如果 d 为 `0`，结果存储在 W 中。如果 d 为 `1`，结果存回寄存器 f（默认）。 f 可以为 256 字节存储区中的任何单元。
如果 a 为 `0`，选择快速操作存储区。如果 a 为 `1`，使用 BSR 选择 GPR 存储区（默认）。
如果 a 为 `0` 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 24.2.3 节 “立即数变址寻址模式中面向字节和面向位的指令”**。

指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写 W

示例： `MOVF REG, 0, 0`

执行指令前
REG = 22h
W = FFh
执行指令后
REG = 22h
W = 22h

MOVFF 将源寄存器的内容移到目标寄存器

语法： MOVFF f_s,f_d

操作数： 0 ≤ f_s ≤ 4095
0 ≤ f_d ≤ 4095

操作： （f_s）→ f_d

受影响的状态位： 无

机器码：

第一个字 （源）	1100	ffff	ffff	$ffff_s$
第二个字 （目标）	1111	ffff	ffff	$ffff_d$

说明： 将源寄存器 f_s 的内容移入目标寄存器 f_d。
源寄存器 f_s 可以是 4096 字节数据空间（000h 到 FFFh）中的任何地址，目标寄存器 f_d 也可以是 000h 到 FFFh 中的任何地址。
源或目标寄存器都可以是 W （这是个有用的特例）。
MOVFF 指令对于将数据存储单元中的内容移入外设寄存器 （如发送缓冲器或 I/O 端口）的场合非常有用。
MOVFF 指令不能使用 PCL、 TOSU、TOSH 或 TOSL 作为目标寄存器。

指令字数： 2

指令周期数： 2 （3）

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 f （源寄存器）	处理数据	空操作
译码	空操作 无效读取	空操作	写 寄存器 f （目标寄存器）

示例： MOVFF REG1, REG2

执行指令前
REG1 = 33h
REG2 = 11h
执行指令后
REG1 = 33h
REG2 = 33h

MOVLB 将立即数移入 BSR 的低半字节

语法： MOVLW k

操作数： 0 ≤ k ≤ 255

操作： k → BSR

受影响的状态位： 无

机器码：

0000	0001	kkkk	kkkk

说明： 将 8 位立即数 k 装入存储区选择寄存器（BSR）。不管 k_7:k_4 的值如何，BSR<7:4> 的值将始终保持为 0。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 立即数 k	处理数据	将立即数 k 写入 BSR

示例： MOVLB 5

执行指令前
BSR 寄存器 = 02h
执行指令后
BSR 寄存器 = 05h

MOVLW 将立即数移入 W

语法：	MOVLW k
操作数：	0 ≤ k ≤ 255
操作：	k → W
受影响的状态位：	无
机器码：	0000 1110 kkkk kkkk
说明：	将 8 位立即数 k 装入 W。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 立即数 k	处理数据	写入 W

示例： `MOVLW 5Ah`

执行指令后
W = 5Ah

MOVWF 将 W 的内容移入 f

语法：	MOVWF f {,a}
操作数：	0 ≤ f ≤ 255 a ∈ [0,1]
操作：	（W）→ f
受影响的状态位：	无
机器码：	0110 111a ffff ffff
说明：	将 W 寄存器中的数据移入寄存器 f。f 可以是 256 字节存储区中的任何地址单元。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。 如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95（5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 24.2.3 节 “立即数变址寻址模式中面向字节和面向位的指令”**。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 f	处理数据	写 寄存器 f

示例： `MOVWF REG, 0`

执行指令前
W = 4Fh
REG = FFh
执行指令后
W = 4Fh
REG = 4Fh

MULLW 将立即数与 W 中的内容相乘

语法：	MULLW k
操作数：	0 ≤ k ≤ 255
操作：	（W）x k → PRODH:PRODL
受影响的状态位：	无
机器码：	`0000` `1101` `kkkk` `kkkk`
说明：	将 W 的内容与 8 位立即数 k 进行无符号的乘法运算。 16 位的结果存储在 PRODH:PRODL 寄存器对中，其中 PRODH 用于存储高字节。 W 的内容不改变。 所有状态标志位都不受影响。 请注意此操作不可能发生溢出或进位。结果有可能为零，但不会反映到相应的标志位。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 立即数 k	处理数据	写 寄存器 PRODH: PRODL

示例： `MULLW 0C4h`

执行指令前
W = E2h
PRODH = ?
PRODL = ?

执行指令后
W = E2h
PRODH = ADh
PRODL = 08h

MULWF 将 W 与 f 的内容相乘

语法：	MULWF f {,a}
操作数：	0 ≤ f ≤ 255 a ∈ [0,1]
操作：	（W）x （f）→ PRODH:PRODL
受影响的状态位：	无
机器码：	`0000` `001a` `ffff` `ffff`
说明：	将 W 的内容与寄存器单元 f 的内容执行无符号的乘法运算。运算的 16 位结果保存在 PRODH:PRODL 寄存器对中，其中 PRODH 存储高字节。 W 和 f 的内容都不改变。 所有状态标志位都不受影响。 请注意此操作不可能发生溢出或进位。结果有可能为零，但不会被检测到。 如果 a 为 `0`，选择快速操作存储区。如果 a 为 `1`，使用 BSR 选择 GPR 存储区 （默认）。 如果 a 为 `0` 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 24.2.3 节 “立即数变址寻址模式中面向字节和面向位的指令”**。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 f	处理数据	写 寄存器 PRODH: PRODL

示例： `MULWF REG, 1`

执行指令前
W = C4h
REG = B5h
PRODH = ?
PRODL = ?

执行指令后
W = C4h
REG = B5h
PRODH = 8Ah
PRODL = 94h

NEGF 对 f 取补

语法：	NEGF f {,a}
操作数：	0 ≤ f ≤ 255 a ∈ [0,1]
操作：	($\overline{f}$) + 1 → f
受影响的状态位：	N、OV、C、DC 和 Z
机器码：	0110 110a ffff ffff
说明：	用二进制补码对单元 f 取补。结果存储在数据存储单元 f 中。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。 如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 24.2.3 节 "立即数变址寻址模式中面向字节和面向位的指令"**。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 f	处理数据	写 寄存器 f

示例： NEGF REG, 1

执行指令前
REG = 0011 1010 [3Ah]
执行指令后
REG = 1100 0110 [C6h]

NOP 空操作

语法：	NOP
操作数：	无
操作：	空操作
受影响的状态位：	无
机器码：	0000 0000 0000 0000 1111 xxxx xxxx xxxx
说明：	不执行任何操作。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	空操作	空操作	空操作

示例：

无。

POP 弹出返回堆栈栈顶的内容

语法：	POP
操作数：	无
操作：	（TOS）→丢弃
受影响的状态位：	无
机器码：	0000 0000 0000 0110
说明：	从返回堆栈弹出 TOS 值并丢弃。然后，前一个压入返回堆栈的值成为 TOS 值。此指令可以让用户正确管理返回堆栈，从而实现软件堆栈。
指令	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	空操作	弹出 TOS 值	空操作

示例：

```
POP
GOTO    NEW
```

执行指令前
- TOS = 0031A2h
- 堆栈 （向下一级） = 014332h

执行指令后
- TOS = 014332h
- PC = NEW

PUSH 将数据压入返回堆栈栈顶

语法：	PUSH
操作数：	无
操作：	（PC）+2 → TOS
受影响的状态位：	无
机器码：	0000 0000 0000 0101
说明：	PC+2 的值被压入返回堆栈的栈顶。原先的 TOS 值被压入堆栈的下一级。此指令允许通过修改 TOS 并将其压入返回堆栈来实现软件堆栈。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	将 PC + 2 压入返回堆栈	空操作	空操作

示例：

```
PUSH
```

执行指令前
- TOS = 345Ah
- PC = 0124h

执行指令后
- PC = 0126h
- TOS = 0126h
- 堆栈 （向下一级） = 345Ah

RCALL 相对调用

语法：	RCALL n
操作数：	-1024 ≤ n ≤ 1023
操作：	（PC）+2 → TOS, （PC）+ 2 + 2n → PC
受影响的状态位：	无
机器码：	1101 1nnn nnnn nnnn
说明：	从当前地址跳转（最多 1K）来调用子程序。首先，将返回地址（PC+2）压入返回堆栈。然后，将二进制补码 “2n” 与 PC 相加。因为 PC 要先递增才能取下一条指令，因此新地址将为 PC + 2 + 2n。该指令为一条双周期指令。
指令字数：	1
指令周期数：	2

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读立即数 n 将 PC 压入堆栈	处理数据	写入 PC
空操作	空操作	空操作	空操作

示例： `HERE RCALL Jump`

执行指令前
PC = 地址（HERE）
执行指令后
PC = 地址（Jump）
TOS = 地址（HERE + 2）

RESET 复位

语法：	RESET
操作数：	无
操作：	将所有受 $\overline{MCLR}$ 复位影响的寄存器和标志位复位。
受影响的状态位：	全部
机器码：	0000 0000 1111 1111
说明：	此指令可实现用软件复位。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	开始复位	空操作	空操作

示例： RESET

执行指令后
寄存器 = 复位值
标志位 * = 复位值

RETFIE 从中断返回

语法：	RETFIE {s}
操作数：	s ∈ [0,1]
操作：	（TOS）→ PC， 1 → GIE/GIEH 或 PEIE/GIEL， 如果 s = 1 （WS）→ W， （STATUSS）→ Status 寄存器， （BSRS）→ BSR， PCLATU 和 PCLATH 保持不变。
受影响的状态位：	GIE/GIEH 和 PEIE/GIEL。
机器码：	0000 0000 0001 000s
说明：	从中断返回。执行出栈操作，将栈顶（TOS）的内容装入 PC。通过将高或低优先级全局中断使能位置 1 ，来使能中断。如果 s = 1，则影子寄存器 WS、STATUSS 和 BSRS 的内容将被装入对应的寄存器 W、Status 和 BSR。如果 s = 0，则不更新这些寄存器 （默认）。
指令字数：	1
指令周期数：	2

Q 周期操作：

Q1	Q2	Q3	Q4
译码	空操作	空操作	从堆栈弹出 PC 值 将 GIEH 或 GIEL 置 1
空操作	空操作	空操作	空操作

示例： RETFIE 1

中断后

PC	=	TOS
W	=	WS
BSR	=	BSRS
Status 寄存器	=	STATUSS
GIE/GIEH, PEIE/GIEL	=	1

RETLW 将立即数返回给 W

语法：	RETLW k
操作数：	0 ≤ k ≤ 255
操作：	k → W， （TOS）→ PC， PCLATU 和 PCLATH 保持不变。
受影响的状态位：	无
机器码：	0000 1100 kkkk kkkk
说明：	将 8 位立即数 k 装入 W。将栈顶内容（返回地址）装入程序计数器。高位地址锁存器 （PCLATH）内容保持不变。
指令字数：	1
指令周期数：	2

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读立即数 “k”	处理数据	从堆栈弹出 PC 值，写入 W
空操作	空操作	空操作	空操作

示例：

```
    CALL  TABLE ; W contains table
                ; offset value
                ; W now has
                ; table value
   :
TABLE
    ADDWF PCL   ; W = offset
    RETLW k0    ; Begin table
    RETLW k1    ;
   :
   :
    RETLW kn    ; End of table
```

执行指令前
W = 07h
执行指令后
W = kn 的值

RETURN 从子程序返回

语法： RETURN {s}

操作数： s ∈ [0,1]

操作： （TOS） → PC，
如果 s = 1
（WS） → W，
（STATUSS） → Status 寄存器，
（BSRS） → BSR，
PCLATU 和 PCLATH 保持不变。

受影响的状态位： 无

机器码：

0000	0000	0001	001s

说明： 从子程序返回。执行出栈操作，将栈顶（TOS）内容装入程序计数器。如果 s = 1，将影子寄存器 WS、 STATUSS 和 BSRS 的内容装入相应的 W、 Status 和 BSR 寄存器。如果 s = 0，则不更新这些寄存器 （默认）。

指令字数： 1

指令周期数： 2

Q 周期操作：

Q1	Q2	Q3	Q4
译码	空操作	处理数据	从堆栈弹出 PC 值
空操作	空操作	空操作	空操作

示例： RETURN

中断后
PC = TOS

RLCF f 带进位循环左移

语法： RLCF f {,d {,a}}

操作数： 0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

操作： （f<n>） → dest<n + 1>，
（f<7>） → C，
（C） → dest<0>

受影响的状态位： C、 N 和 Z

机器码：

0011	01da	ffff	ffff

说明： 将寄存器 f 的内容连同进位标志位一起循环左移 1 位。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f （默认）。
如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区 （默认）。
如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 24.2.3 节 “立即数变址寻址模式中面向字节和面向位的指令”**。

C ← 寄存器 f ←

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入目标寄存器

示例： RLCF REG, 0, 0

执行指令前
REG = 1110 0110
C = 0
执行指令后
REG = 1110 0110
W = 1100 1100
C = 1

RLNCF　f 循环左移 （不带进位）

语法：	RLNCF　f {,d {,a}}
操作数：	0 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]
操作：	（f<n>）→ dest<n + 1>， （f<7>）→ dest<0>
受影响的状态位：	N 和 Z
机器码：	0100 01da ffff ffff
说明：	将寄存器 f 的内容循环左移 1 位。 如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f （默认）。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。 如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 24.2.3 节 “立即数变址寻址模式中面向字节和面向位的指令”**。
指令字数：	1
指令周期数：	1

寄存器 f

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 f	处理数据	写入 目标寄存器

示例：　RLNCF　REG, 1, 0

执行指令前
REG = 1010 1011
执行指令后
REG = 0101 0111

RRCF　f 带进位循环右移

语法：	RRCF　f {,d {,a}}
操作数：	0 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]
操作：	（f<n>）→ dest<n – 1>， （f<0>）→ C， （C）→ dest<7>
受影响的状态位：	C、 N 和 Z
机器码：	0011 00da ffff ffff
说明：	将寄存器 f 的内容连同进位标志位一起循环右移 1 位。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f （默认）。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。 如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 24.2.3 节“立即数变址寻址模式中面向字节和面向位的指令”** .
指令字数：	1
指令周期数：	1

C　寄存器 f

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 f	处理数据	写入 目标寄存器

示例：　RRCF　REG, 0, 0

执行指令前
REG = 1110 0110
C = 0
执行指令后
REG = 1110 0110
W = 0111 0011
C = 0

RRNCF　f 循环右移 （不带进位）

语法：　RRNCF　f {,d {,a}}

操作数：　0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

操作：　（f<n>） → dest<n – 1>，
（f<7>） → dest<0>

受影响的状态位：　N 和 Z

机器码：

0100	00da	ffff	ffff

说明：　将寄存器 f 的内容循环右移 1 位。 如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f （默认）。
如果 a 为 0，选择快速操作存储区。忽略 BSR 的值。如果 a 为 1，使用 BSR 选择 GPR 存储区 （默认）。
如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 24.2.3 节“立即数变址寻址模式中面向字节和面向位的指令”**。

寄存器 f

指令字数：　1

指令周期数：　1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 f	处理数据	写入 目标寄存器

例 1：　RRNCF　REG, 1, 0

执行指令前
REG　=　1101 0111
执行指令后
REG　=　1110 1011

例 2：　RRNCF　REG, 0, 0

执行指令前
W　=　?
REG　=　1101 0111
执行指令后
W　=　1110 1011
REG　=　1101 0111

SETF　将 f 的内容置为全 1

语法：　SETF　f {,a}

操作数：　0 ≤ f ≤ 255
a ∈ [0,1]

操作：　FFh → f

受影响的状态位：　无

机器码：

0110	100a	ffff	ffff

说明：　将指定寄存器的内容置为 FFh。
如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区 （默认）。
如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 24.2.3 节“立即数变址寻址模式中面向字节和面向位的指令”**。

指令字数：　1

指令周期数：　1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 f	处理数据	写 寄存器 f

示例：　SETF　REG,1

执行指令前
REG　=　5Ah
执行指令后
REG　=　FFh

SLEEP 进入休眠模式

语法：	SLEEP
操作数：	无
操作：	00h → WDT， 0 → WDT 后分频器， 1 → $\overline{\text{TO}}$， 0 → $\overline{\text{PD}}$
受影响的状态位：	$\overline{\text{TO}}$ 和 $\overline{\text{PD}}$
机器码：	0000 0000 0000 0011
说明：	掉电状态位 （$\overline{\text{PD}}$）清零。超时状态位（$\overline{\text{TO}}$）置 1。看门狗定时器及其后分频器清零。 振荡器停振，处理器进入休眠模式。
指令字数：	1
指令周期数：	1

Q 周期操作：

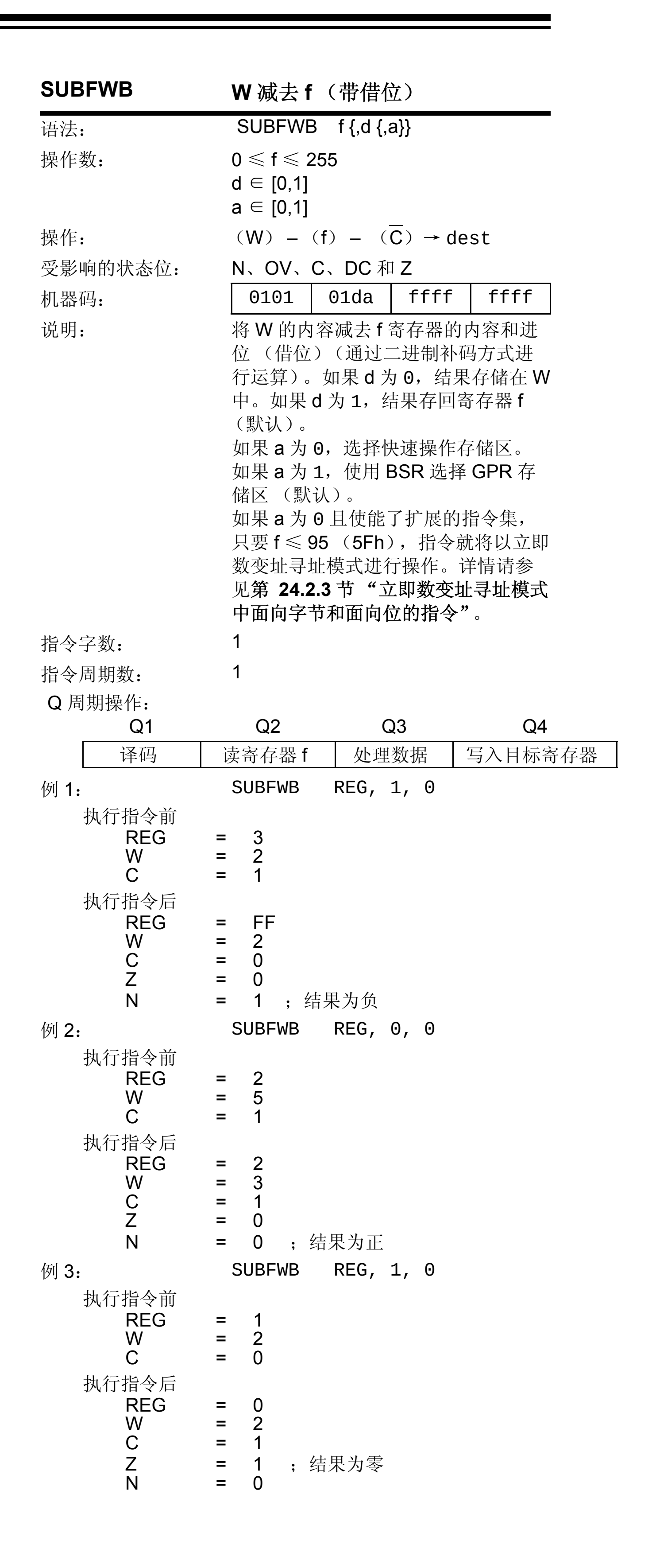
Q1	Q2	Q3	Q4
译码	空操作	处理数据	进入休眠模式

示例： SLEEP

执行指令前
$\overline{\text{TO}}$ = ?
$\overline{\text{PD}}$ = ?

执行指令后
$\overline{\text{TO}}$ = 1 †
$\overline{\text{PD}}$ = 0

† 如果由 WDT 引起唤醒，则此位将被清零。



SUBFWB W 减去 f （带借位）

语法：	SUBFWB f {,d {,a}}
操作数：	0 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]
操作：	（W） – （f） – （$\overline{\text{C}}$） → dest
受影响的状态位：	N、OV、C、DC 和 Z
机器码：	0101 01da ffff ffff
说明：	将 W 的内容减去 f 寄存器的内容和进位 （借位）（通过二进制补码方式进行运算）。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f （默认）。 如果 a 为 0，选择快速操作存储区。 如果 a 为 1，使用 BSR 选择 GPR 存储区 （默认）。 如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见第 **24.2.3** 节 **"立即数变址寻址模式中面向字节和面向位的指令"**。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入目标寄存器

例 1： SUBFWB REG, 1, 0

执行指令前
REG = 3
W = 2
C = 1

执行指令后
REG = FF
W = 2
C = 0
Z = 0
N = 1 ；结果为负

例 2： SUBFWB REG, 0, 0

执行指令前
REG = 2
W = 5
C = 1

执行指令后
REG = 2
W = 3
C = 1
Z = 0
N = 0 ；结果为正

例 3： SUBFWB REG, 1, 0

执行指令前
REG = 1
W = 2
C = 0

执行指令后
REG = 0
W = 2
C = 1
Z = 1 ；结果为零
N = 0

 初稿

SUBLW 立即数减去 W 的内容

语法：	SUBLW k
操作数：	0 ≤ k ≤ 255
操作：	k –（W）→ W
受影响的状态位：	N、OV、C、DC 和 Z
机器码：	0000 | 1000 | kkkk | kkkk
说明：	用 8 位立即数 k 减去 W。结果保存在 W 寄存器中。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读立即数 k	处理数据	写入 W

例 1： `SUBLW 02h`

```
执行指令前
    W    =  01h
    C    =  ?
执行指令后
    W    =  01h
    C    =  1   ；结果为正
    Z    =  0
    N    =  0
```

例 2： `SUBLW 02h`

```
执行指令前
    W    =  02h
    C    =  ?
执行指令后
    W    =  00h
    C    =  1   ；结果为零
    Z    =  1
    N    =  0
```

例 3： `SUBLW 02h`

```
执行指令前
    W    =  03h
    C    =  ?
执行指令后
    W    =  FFh ；（2 进制补码）
    C    =  0   ；结果为负
    Z    =  0
    N    =  1
```

SUBWF f 减去 W

语法：	SUBWF f {,d {,a}}
操作数：	0 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]
操作：	（f） –（W）→ dest
受影响的状态位：	N、OV、C、DC 和 Z
机器码：	0101 | 11da | ffff | ffff
说明：	用寄存器 f 中的内容减去 W 寄存器的内容 （通过二进制补码方式进行运算）。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 “f” （默认）。 如果 a 为 0，选择快速操作存储区。 如果 a 为 1，使用 BSR 选择 GPR 存储区 （默认）。 如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见第 **24.2.3 节 “立即数变址寻址模式中面向字节和面向位的指令”**。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入目标寄存器

例 1： `SUBWF REG, 1, 0`

```
执行指令前
    REG  =  3
    W    =  2
    C    =  ?
执行指令后
    REG  =  1
    W    =  2
    C    =  1   ；结果为正
    Z    =  0
    N    =  0
```

例 2： `SUBWF REG, 0, 0`

```
执行指令前
    REG  =  2
    W    =  2
    C    =  ?
执行指令后
    REG  =  2
    W    =  0
    C    =  1   ；结果为零
    Z    =  1
    N    =  0
```

例 3： `SUBWF REG, 1, 0`

```
执行指令前
    REG  =  1
    W    =  2
    C    =  ?
执行指令后
    REG  =  FFh ；（二进制补码）
    W    =  2
    C    =  0   ；结果为负
    Z    =  0
    N    =  1
```

 初稿

SUBWFB　f 减去 W （带借位）

语法：	SUBWFB　f {,d {,a}}
操作数：	0 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]
操作：	（f）–（W）–（$\overline{C}$）→ dest
受影响的状态位：	N、OV、C、DC 和 Z
机器码：	0101 / 10da / ffff / ffff
说明：	用 f 寄存器的内容减去 W 的内容和进位（借位）（通过二进制补码方式进行运算）。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f （默认）。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。 如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 24.2.3 节"立即数变址寻址模式中面向字节和面向位的指令"**。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入目标寄存器

例 1：　SUBWFB　REG, 1, 0

执行指令前
REG = 19h （0001 1001）
W = 0Dh （0000 1101）
C = 1

执行指令后
REG = 0Ch （0000 1011）
W = 0Dh （0000 1101）
C = 1
Z = 0
N = 0 ；结果为正

例 2：　SUBWFB　REG, 0, 0

执行指令前
REG = 1Bh （0001 1011）
W = 1Ah （0001 1010）
C = 0

执行指令后
REG = 1Bh （0001 1011）
W = 00h
C = 1
Z = 1 ；结果为零
N = 0

例 3：　SUBWFB　REG, 1, 0

执行指令前
REG = 03h （0000 0011）
W = 0Eh （0000 1101）
C = 1

执行指令后
REG = F5h （1111 0100） ；【2 进制补码】
W = 0Eh （0000 1101）
C = 0
Z = 0
N = 1 ；结果为负

SWAPF　将 f 的高半字节和低半字节交换

语法：	SWAPF　f {,d {,a}}
操作数：	0 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]
操作：	（f<3:0>）→ dest<7:4>， （f<7:4>）→ dest<3:0>
受影响的状态位：	无
机器码：	0011 / 10da / ffff / ffff
说明：	将寄存器 f 的高半字节和低半字节互相交换。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f （默认）。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。 如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 24.2.3 节"立即数变址寻址模式中面向字节和面向位的指令"**。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入目标寄存器

示例：　SWAPF　REG, 1, 0

执行指令前
REG = 53h

执行指令后
REG = 35h

TBLRD 表读

语法：	TBLRD （ *; *+; *-; +*）
操作数：	无
操作：	如果执行 TBLRD *， （程序存储器 （TBLPTR）） → TABLAT； TBLPTR 不改变。 如果执行 TBLRD *+， （程序存储器 （TBLPTR）） → TABLAT； （TBLPTR） + 1 → TBLPTR。 如果执行 TBLRD *-， （程序存储器 （TBLPTR）） → TABLAT； （TBLPTR） – 1 → TBLPTR。 如果执行 TBLRD +*， （TBLPTR） + 1 → TBLPTR； （程序存储器 （TBLPTR）） → TABLAT。
受影响的状态位：	无

机器码：

0000	0000	0000	10nn nn = 0 * = 1 *+ = 2 *- = 3 +*

说明：

该指令用于读取程序存储器 （P.M.）的内容。使用表指针 （TBLPTR）对程序存储器进行寻址。

TBLPTR （一个 21 位指针）指向程序存储器的每个字节。 TBLPTR 的寻址范围为 2 MB。

TBLPTR[0] = 0：程序存储器字的低有效字节

TBLPTR[0] = 1：程序存储器字的高有效字节

TBLRD 指令可用如下方法修改 TBLPTR 的值：

- 不变
- 后加
- 后减
- 预加

指令字数： 1

指令周期数： 2

Q 周期操作：

Q1	Q2	Q3	Q4
译码	空操作	空操作	空操作
空操作	空操作 （读程序存储器）	空操作	空操作 （写 TABLAT）

TBLRD 表读 （续）

例 1： `TBLRD *+ ;`

执行指令前

TABLAT	=	55h
TBLPTR	=	00A356h
存储单元 （00A356h）	=	34h

执行指令后

TABLAT	=	34h
TBLPTR	=	00A357h

例 2： `TBLRD +* ;`

执行指令前

TABLAT	=	0AAh
TBLPTR	=	01A357h
存储单元 （01A357h）	=	12h
存储单元 （01A358h）	=	34h

执行指令后

TABLAT	=	34h
TBLPTR	=	01A358h

TBLWT 表写

语法： TBLWT （ *; *+; *-; +*）

操作数： 无

操作： 如果执行 TBLWT*，
（TABLAT）→保持寄存器；
TBLPTR 不改变；
如果执行 TBLWT*+，
（TABLAT）→ 保持寄存器；
（TBLPTR）+ 1 → TBLPTR ；
如果执行 TBLWT*-，
（TABLAT）→保持寄存器；
（TBLPTR）– 1 → TBLPTR ；
如果执行 TBLWT+*，
（TBLPTR）+ 1 → TBLPTR ；
（TABLAT）→保持寄存器；

受影响的状态位：无

机器码：

0000	0000	0000	11nn nn = 0 * = 1 *+ = 2 *- = 3 +*

说明： 此指令使用 TBLPTR 的低 3 位来确定要将 TABLAT 中的内容写入 8 个保持寄存器中的哪一个。该保持寄存器用于对程序存储器（P.M.）的内容编程。（关于对闪存程序存储器编程的更多详情，请参见**第 6.0 节 " 闪存程序存储器 "**。）
TBLPTR （一个 21 位指针）指向程序存储器的每个字节。 TBLPTR 的寻址范围为 2 MB。 TBLPTR 的 LSb 选择要访问的程序存储器单元。

TBLPTR[0] = 0：程序存储器字的最低有效字节

TBLPTR[0] = 1：程序存储器字的最高有效字节

TBLWT 指令可用如下方式修改 TBLPTR 的值：
- 不变
- 后加
- 后减
- 预加

指令字数： 1

指令周期数： 2

Q 周期操作：

Q1	Q2	Q3	Q4
译码	空操作	空操作	空操作
空操作	空操作 （读 TABLAT）	空操作	空操作 （写保持寄存器）

TBLWT 表写 （续）

例 1： TBLWT *+;

执行指令前

TABLAT	=	55h
TBLPTR	=	00A356h
保持寄存器（00A356h）	=	FFh

执行指令后 （表写操作完成）

TABLAT	=	55h
TBLPTR	=	00A357h
保持寄存器（00A356h）	=	55h

例 2： TBLWT +*;

执行指令前

TABLAT	=	34h
TBLPTR	=	01389Ah
保持寄存器（01389Ah）	=	FFh
保持寄存器（01389Bh）	=	FFh

执行指令后 （表写操作完成）

TABLAT	=	34h
TBLPTR	=	01389Bh
保持寄存器（01389Ah）	=	FFh
保持寄存器（01389Bh）	=	34h

> **注：** 表写（TBLWT）指令在 PIC18F6X90/8X90 器件的用户模式不可用，因为这类器件是没有外部总线接口的标准闪存器件。

TSTFSZ　测试 f，为 0 则跳过

语法：	TSTFSZ f {,a}
操作数：	0 ≤ f ≤ 255 a ∈ [0,1]
操作：	f 为 0 则跳过
受影响的状态位：	无
机器码：	0110 011a ffff ffff
说明：	如果 f = 0，丢弃已取的指令并执行一条 NOP 指令，使这条指令成为双周期指令。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。 如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95（5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 24.2.3 节“立即数变址寻址模式中面向字节和面向位的指令”**。
指令字数：	1
指令周期数：	1（2） **注：**如果跳过的指令后跟有一条双字指令，则执行 TSTFSZ 需要 3 个周期。

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 f	处理数据	空 操作

如果跳过：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作

如果跳过的指令后面跟有 2 字指令：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作
空操作	空操作	空操作	空操作

示例：

```
HERE    TSTFSZ  CNT, 1
NZERO   :
ZERO    :
```

执行指令前
　PC = 地址（HERE）

执行指令后
　如果 CNT = 00h，
　PC = 地址（ZERO）
　如果 CNT ≠ 00h，
　PC = 地址（NZERO）

XORLW　立即数与 W 作逻辑异或运算

语法：	XORLW k
操作数：	0 ≤ k ≤ 255
操作：	（W）.XOR. k → W
受影响的状态位：	N 和 Z
机器码：	0000 1010 kkkk kkkk
说明：	将 W 的内容与 8 位立即数 k 进行逻辑异或运算。结果保存在 W 寄存器中。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 立即数 k	处理数据	写入 W

示例：

```
XORLW   0AFh
```

执行指令前
　W = B5h

执行指令后
　W = 1Ah

XORWF　W 与 f 作逻辑异或运算

语法：	XORWF　f {,d {,a}}
操作数：	0 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]
操作：	（W）.XOR.（f）→ dest
受影响的状态位：	N 和 Z
机器码：	0001 | 10da | ffff | ffff
说明：	将 W 的内容与寄存器 f 的内容进行逻辑异或运算。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f （默认）。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。 如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 24.2.3 节“立即数变址寻址模式中面向字节和面向位的指令”**。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 f	处理数据	写入 目标寄存器

示例：　XORWF　REG, 1, 0

执行指令前
REG　=　AFh
W　=　B5h

执行指令后
REG　=　1Ah
W　=　B5h

24.2 扩展的指令集

除了PIC18指令集的75个标准指令之外，PIC18F6390/6490/8390/8490 器件还提供针对内核 CPU 功能的可选扩展指令。这些新增的功能包括 8 个额外的指令，它们可以实现间接和变址寻址操作以及对许多标准PIC18指令执行立即数变址寻址。

这些额外功能在默认情况下是禁止的。用户必须通过将 XINST 配置位置 1，才能使能它们。

扩展指令集中的指令可以全部被归为立即数操作类指令，它们既可以控制文件选择寄存器也可以使用这些寄存器进行变址寻址。 其中的两个指令 ADDFSR 和 SUBFSR，可以直接对 FSR2 进行操作，而 ADDULNK 和 SUBULNK 指令允许在执行后自动返回。

这些扩展的指令专门用于优化用高级语言特别是C语言编写的重入程序代码（也就是递归调用或使用软件堆栈的代码）。此外，它们使用户能更有效地用高级语言对数据结构执行特定的操作。这些操作包括：

- 在进入和退出子程序时对软件堆栈空间进行动态分配和释放。
- 功能指针调用
- 对软件堆栈指针进行控制
- 对软件堆栈中的变量进行控制

表 24-3提供了扩展指令集中的指令汇总。第 **24.2.2** 节 **"扩展的指令集"** 对这些指令的进行了详细说明。表 24-1 提供了标准和扩展的 PIC18 指令集的操作码字段说明。

> **注：** 扩展的指令集和立即数变址寻址模式是专为优化用 C 语言编写的应用程序而设计的，用户可能不会在汇编器中直接使用这些指令。对于那些查看编译器生成代码的用户，这些命令的语法可作为参考。

24.2.1 扩展指令的语法

大部分扩展指令都使用变址参数，同时使用一个文件选择寄存器和某一偏移量来指定源寄存器或目标寄存器。当指令的参数作为变址寻址的一部分时，会用方括号

（"[ ]"）把它括起来。这时表示此参数用作变址地址或偏移量。如果 MPASM™ 汇编器发现一个变址地址或偏移量没有被括起来，它就会标出一个错误。

当使能扩展的指令集时，括号也用于表示面向字节和面向位的指令中的变址参数。这是对指令语法的额外更改。欲知更多信息，请参见**第 24.2.3.1 节 " 标准 PIC18 命令的扩展指令语法 "**。

> **注：** 以前，在 PIC18 和早期的指令集中使用方括号来表示可选参数。在此文本和以后的文本中，可选参数将用大扩号（"{ }"）表示。

表 24-3： PIC18 指令集的扩展

助记符，操作数		说明	周期	16 位指令字 MSb			LSb	受影响的状态位
ADDFSR	f, k	将立即数与 FSR 相加	1	1110	1000	ffkk	kkkk	无
ADDULNK	k	将立即数与 FSR2 相加并返回	2	1110	1000	11kk	kkkk	无
CALLW		使用 WREG 调用子程序	2	0000	0000	0001	0100	无
MOVSF	z_s, f_d	将 z_s （源）（第一个字） 移入 f_d （目标）（第二个字）	2	1110 1111	1011 ffff	0zzz ffff	zzzz ffff	无
MOVSS	z_s, z_d	将 z_s （源）（第一个字） 移入 z_d （目标）（第二个字）	2	1110 1111	1011 xxxx	1zzz xzzz	zzzz zzzz	无
PUSHL	k	将立即数保存在 FSR2 FSR2 减 1	1	1110	1010	kkkk	kkkk	无
SUBFSR	f, k	FSR 减去立即数	1	1110	1001	ffkk	kkkk	无
SUBULNK	k	FSR2 减去立即数并返回	2	1110	1001	11kk	kkkk	无

24.2.2 扩展的指令集

ADDFSR **FSR 的内容与立即数相加**

语法： ADDFSR f, k

操作数： $0 \leqslant k \leqslant 63$
$f \in [0, 1, 2]$

操作： FSR （f）+ k → FSR （f）

受影响的状态位： 无

机器码：

1110	1000	ffkk	kkkk

说明： 将由 f 指定的 FSR 的内容加上一个 6 位的立即数 k。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读立即数 k	处理数据	写入 FSR

示例： `ADDFSR 2, 23h`

执行指令前
FSR2 = 03FFh
执行指令后
FSR2 = 0422h

ADDULNK **FSR2 的内容与立即数相加并返回**

语法： ADDULNK k

操作数： $0 \leqslant k \leqslant 63$

操作： FSR2 + k → FSR2,
PC = （TOS）

受影响的状态位： 无

机器码：

1110	1000	11kk	kkkk

说明： 将 FSR2 的内容加上一个 6 位的立即数 k。然后通过将 TOS 的值装入 PC，执行一条 RETURN 指令。

执行该指令需要两个周期；在第 2 个周期执行一条 NOP 指令。

该指令可以被认为是 ADDFSR 指令的特例，其中 f = 3 （二进制 “11”），它仅针对 FSR2 进行操作。

指令字数： 1

指令周期数： 2

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读立即数 k	处理数据	写入 FSR
空操作	空操作	空操作	空操作

示例： `ADDULNK 23h`

执行指令前
FSR2 = 03FFh
PC = 0100h
执行指令后
FSR2 = 0422h
PC = （TOS）

注： 所有的 PIC18 指令都可能在其指令助记符之前使用可选的标号参数，用于符号寻址。如果使用了标号，那么指令语法将变为：{label} 指令参数。

CALLW 使用 **WREG** 调用子程序

语法： CALLW

操作数： 无

操作： （PC + 2） → TOS，
（W） → PCL，
（PCLATH） → PCH，
（PCLATU） → PCU

受影响的状态位： 无

机器码：

0000	0000	0001	0100

说明： 首先，返回地址 （PC+2）被压入返回堆栈。接下来，将 W 寄存器的内容写入 PCL， PCL 现有的值被丢弃。然后， PCLATH 和 PCLATU 的内容被分别锁存到 PCH 和 PCU。第二个周期执行一条 NOP 指令，并同时取指下一条指令。
和 CALL 不一样，该指令没有更新 W、 Status 或 BSR 寄存器的选项。

指令字数： 1

指令周期数： 2

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 WREG	将 PC 压入堆栈	空操作
空操作	空操作	空操作	空操作

示例： `HERE    CALLW`

执行指令前
- PC = 地址 （HERE）
- PCLATH = 10h
- PCLATU = 00h
- W = 06h

执行指令后
- PC = 001006h
- TOS = 地址 （HERE + 2）
- PCLATH = 10h
- PCLATU = 00h
- W = 06h

MOVSF 将变址寻址单元内容移入 **f**

语法： MOVSF [z_s], f_d

操作数： 0 ≤ zs ≤ 127
0 ≤ fd ≤ 4095

操作： （（FSR2） + z_s） → f_d

受影响的状态位： 无

机器码：

第一个字 （源）	1110	1011	0zzz	zzzzs
第二个字 （目标）	1111	ffff	ffff	ffffd

说明： 将源寄存器的内容移入目标寄存器 f_d。通过将第一个字中的 7 位立即数偏移量 z_s 与 FSR2 的值相加来确定源寄存器的实际地址。第二个字中的 12 位立即数 f_d 指向目标寄存器的地址。两个地址均可以是 4096 字节的数据空间 （000h 到 FFFh）中的任何单元。
MOVFF 指令中的目标寄存器不能是 PCL、 TOSU、 TOSH 或 TOSL。
如果计算得到的源地址指向间接寻址寄存器，将返回 00h。

指令字数： 2

指令周期数： 2

Q 周期操作：

Q1	Q2	Q3	Q4
译码	确定源地址	确定源地址	读源寄存器
译码	空操作 无效读取	空操作	写目标寄存器 f

示例： `MOVSF   [05h], REG2`

执行指令前
- FSR2 = 80h
- 85h 单元的内容 = 33h
- REG2 = 11h

执行指令后
- FSR2 = 80h
- 85h 单元的内容 = 33h
- REG2 = 33h

MOVSS 变址寻址移动数据

语法：	MOVSS [z_s], [z_d]
操作数：	$0 \leqslant z_s \leqslant 127$ $0 \leqslant z_d \leqslant 127$
操作：	（（FSR2）+ z_s）→（（FSR2）+ z_d）
受影响的状态位：	无

机器码：

第一个字 （源）	1110	1011	1zzz	$zzzz_s$
第二个字 （目标）	1111	xxxx	xzzz	$zzzz_d$

说明： 将源寄存器的内容移到目标寄存器。通过将 FSR2 中的值分别加上 7 位立即数偏移量 z_s 或 z_d 来确定源寄存器和目标寄存器的地址。两个寄存器都可以是 4096 字节数据存储器空间 （000h 到 FFFh）中的任意单元。
MOVSS 指令不能使用 PCL、 TOSU、 TOSH 或 TOSL 作为目标寄存器。
如果计算得到的源地址指向间接寻址寄存器，将返回 00h。如果计算得到的目标地址指向间接寻址寄存器，将执行一条 NOP 指令。 .

指令字数： 2

指令周期数： 2

Q 周期操作：

Q1	Q2	Q3	Q4
译码	确定源地址	确定源地址	读源寄存器
译码	确定 目标地址	确定 目标地址	写 目标寄存器

示例： MOVSS [05h], [06h]

执行指令前
FSR2 = 80h
85h 单元的内容 = 33h
86h 单元的内容 = 11h

执行指令后
FSR2 = 80h
85h 单元的内容 = 33h
86h 单元的内容 = 33h

PUSHL 将立即数保存在 **FSR2**，**FSR2** 减 **1**

语法：	PUSHL k
操作数：	$0 \leqslant k \leqslant 255$
操作：	k →（FSR2）， FSR2 – 1 → FSR2
受影响的状态位：	无

机器码：

1111	1010	kkkk	kkkk

说明： 8 位立即数 k 被写入由 FSR2 指定的数据存储器单元。操作完后 FSR2 减 1。
此指令允许用户将值压入软件堆栈。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读取 k	处理 数据	写入 目标寄存器

示例： PUSHL 08h

执行指令前
FSR2H:FSR2L = 01ECh
存储单元 （01ECh） = 00h

执行指令后
FSR2H:FSR2L = 01EBh
存储单元 （01ECh） = 08h

SUBFSR FSR 减去立即数

语法：	SUBFSR f, k
操作数：	0 ≤ k ≤ 63 f∈ [0, 1, 2]
操作：	FSRf – k → FSRf
受影响的状态位：	无
机器码：	`1110` `1001` `ffkk` `kkkk`
说明：	用 f 指定的 FSR 的内容减去 6 位立即数 k。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 f	处理数据	写 目标寄存器

示例：　　`SUBFSR 2, 23h`

执行指令前
　FSR2　=　03FFh
执行指令后
　FSR2　=　03DCh

SUBULNK FSR2 减去立即数并返回

语法：	SUBULNK　k
操作数：	0 ≤ k ≤ 63
操作：	FSR2 – k → FSR2 （TOS） →PC
受影响的状态位：	无
机器码：	`1110` `1001` `11kk` `kkkk`
说明：	用 FSR 的内容减去 6 位立即数 k，然后通过将 TOS 装入 PC 执行一条 `RETURN` 指令。执行该指令需要两个指令周期，第二个指令周期执行一条 `NOP` 指令。 该指令是 `SUBFSR` 指令的特殊情况，其中 f = 3 （二进制数 “`11`”）。它只针对 FSR2 进行操作。
指令字数：	1
指令周期数：	2

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 f	处理数据	写 目标寄存器
空操作	空操作	空操作	空操作

示例：　　`SUBULNK 23h`

执行指令前
　FSR2　=　03FFh
　PC　=　0100h
执行指令后
　FSR2　=　03DCh
　PC　=　（TOS）

24.2.3 立即数变址寻址模式中面向字节和面向位的指令

注：	使能 PIC18 扩展指令集可能导致常规应用程序运行不正常或完全失败。

一旦使能扩展的指令集，除了可以使用 8 条新命令之外，还可以使用立即数变址寻址模式（第 **5.6.1 节 "使用立即数偏移量进行变址寻址"**）。这将导致标准PIC18 指令的地址解析方法有很大变化。

当禁用扩展的指令集时，嵌入在操作码中的地址被视作立即数存储单元：可以是快速操作存储区中的单元 （a = 0），或由 BSR 指定的 GPR 存储区中的单元 （a = 1）。当使能扩展的指令集且 a = 0 时，地址为 5Fh 或以下的文件寄存器参数被解析为 FSR2 中的指针值的偏移量，而不是一个立即数地址。对于实际应用来说，这意味着所有使用快速操作 RAM 位作为参数的指令，即所有面向字节或面向位的指令，或者几乎半数的 PIC18 内核指令，在使能了扩展的指令集时操作都会有所不同。

当 FSR2 的内容为 00h 时，快速操作 RAM 的边界会被重新映射到它们的原始值。这对于编写向下兼容的代码很有用处。如果使用此技术，有必要在 "C" 程序调用汇编子程序时保存 FSR2 的值并在返回时将它恢复，这样做的目的是保护堆栈指针。用户还必须记住扩展指令集的语法要求（见第 **24.2.3.1 节 "标准 PIC18 命令的扩展指令语法 "**）。

虽然立即数变址寻址模式对于动态堆栈和指针控制很有用处，但是如果不小心误用了寄存器也会非常麻烦。已经习惯使用 PIC18 编程的用户必须记住，在使能了扩展的指令集后，地址小于或等于 5Fh 的寄存器用于立即数变址寻址。

下面是在立即数变址寻址模式中，一些面向字节和位的指令的示例，通过示例可以看出指令执行如何受到影响。示例中的操作数条件适用于所有这一类的指令。

24.2.3.1 标准 PIC18 命令的扩展指令语法

当使能了扩展的指令集时，立即数偏移量 "k" 被用来替换标准的面向字节和位的命令中的文件寄存器参数 "f"。如前所述，只有在 "f" 小于或等于 5Fh 时才会发生这种情况。当使用偏移量时，该偏移量必须用方括号 "[]" 标出。因为在扩展的指令集中，编译器将括号中的数值解析为变址地址或偏移量。省略括号，或在括号内使用大于 5Fh 的值会在 MPASM™ 汇编器中产生错误。

如果变址参数已被加上了括号，那么就不再需要指定快速操作 RAM 参数；此参数被假定为 0。这与标准操作（禁止扩展的指令集时）刚好相反。在变址寻址模式中，定义快速操作 RAM 位也将在 MPASM 汇编器中产生错误。

目标参数 "d" 的操作和以前一样。

在 MPASM 汇编器的最新版本中，必须明确调用对扩展的指令集的语言支持。可以通过命令行选项 /y 或在源代码中加入 PE 伪指令进行调用。

24.2.4 使能扩展指令集时的注意事项

需要注意的是并非所有用户都有必要使用扩展的指令集，尤其是那些不使用软件堆栈的用户。

此外，立即数变址寻址模式可能会对写入 PIC18 汇编器的常规应用程序带来问题。这是因为常规的指令会尝试寻址快速操作存储区中地址低于 5Fh 的寄存器。当使能了扩展的指令集时，这些地址被解析为相对于 FSR2 的立即数偏移量，所以应用程序会读或写错误的地址。

将应用程序移植到 PIC18FXX90 器件时，代码的类型是非常重要的。在使用扩展的指令集时，用 C 语言编写的代码较长的重入应用程序会运行的很好，而大量使用快速操作存储区的常规应用程序不会获得任何益处。

ADDWF 将 W 与变址寻址单元的内容相加（立即数变址寻址模式）

语法：	ADDWF [k] {,d}
操作数：	0 ≤ k ≤ 95 d ∈ [0,1] a = 0
操作：	（W）+ （（FSR2）+ k）→ dest
受影响的状态位：	N、 OV、C、DC 和 Z
机器码：	0010 01d0 kkkk kkkk
说明：	将 W 的内容与由 FSR2 加上偏移量 k” 指定的寄存器的内容相加。 如果 d 为 0，结果被存储在 W 中。如果 d 为 1，结果存回寄存器 f（默认）。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读取 k	处理数据	写入目标寄存器

示例： ADDWF [OFST] ,0

执行指令前
- W = 17h
- OFST = 2Ch
- FSR2 = 0A00h
- 0A2Ch 单元的内容 = 20h

执行指令后
- W = 37h
- 0A2Ch 单元的内容 = 20h

BSF 将变址寻址单元相应位置 1 （立即数变址寻址模式）

语法：	BSF [k], b
操作数：	0 ≤ f ≤ 95 0 ≤ b ≤ 7 a = 0
操作：	1 → （（FSR2 + k）） <b>
受影响的状态位：	无
机器码：	1000 bbb0 kkkk kkkk
说明：	将由 FSR2 加上偏移量 k 指定的寄存器中的位 b 置 1。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入目标寄存器

示例： BSF [FLAG_OFST], 7

执行指令前
- FLAG_OFST = 0Ah
- FSR2 = 0A00h
- 0A0Ah 单元的内容 = 55h

执行指令后
- 0A0Ah 单元的内容 = D5h

SETF 将变址寻址单元置全 1 （立即数变址寻址模式）

语法：	SETF [k]
操作数：	0 ≤ k ≤ 95
操作：	FFh → （（FSR2）+ k）
受影响的状态位：	无
机器码：	0110 1000 kkkk kkkk
说明：	将由 FSR2 加上偏移量 “k” 指定的寄存器的内容置为 FFh。
指令字数：	1
指令周期数：	1

示例： SETF [OFST]

执行指令前
- OFST = 2Ch
- FSR2 = 0A00h
- 0A2Ch 单元的内容 = 00h

执行指令后
- 0A2Ch 单元的内容 = FFh

24.2.5 使用 MICROCHIP MPLAB® IDE 工具的注意事项

最新版本的 Microchip 软件工具完全支持 PIC18F6390/6490/8390/8490 系列器件的扩展指令集。包括 MPLAB C18 C 语言编译器、 MPASM 汇编语言和 MPLAB 集成开发环境 （IDE）

在选择了使用软件开发的目标器件后， MPLAB IDE 将把该器件的默认配置位自动置 1。在禁用扩展的指令集和立即数变址寻址模式时，XINST配置位的默认设置是 0。在编程过程中必须将 XINST 位置 1 才能正确地利用扩展的指令集开发应用程序。

要使用扩展的指令集开发软件，用户必须设置他们的语言工具以实现对扩展指令和变址寻址模式的支持。根据所使用的环境，可以通过以下几种方法：

- 开发环境中的菜单选项或对话框，允许用户配置项目的语言工具及其设置
- 命令行选项
- 源代码中的伪指令

这些选项在不同的编译器、汇编器和开发环境中将有所不同。建议用户在其开发系统所附带的文档中查询相应的信息。

25.0 开发支持

一系列硬件及软件开发工具对 PICmicro® 单片机提供支持：

- 集成开发环境
  - MPLAB® IDE 软件
- 汇编器 / 编译器 / 链接器
  - MPASM™ 汇编器
  - MPLAB C18 和 MPLAB C30 C 编译器
  - MPLINK™ 目标链接器 / MPLIB™ 目标库管理器
  - MPLAB ASM30 汇编器 / 链接器 / 库
- 模拟器
  - MPLAB SIM 软件模拟器
- 仿真器
  - MPLAB ICE 2000 在线仿真器
  - MPLAB ICE 4000 在线仿真器
- 在线调试器
  - MPLAB ICD 2
- 器件编程器
  - PICSTART® Plus 开发编程器
  - MPLAB PM3 器件编程器
- 低成本演示和开发板及评估工具包

25.1 MPLAB 集成开发环境软件

MPLAB IDE 软件为 8/16 位单片机市场提供了前所未有的易于使用的软件开发平台。MPLAB IDE 是基于 Windows® 操作系统的应用软件，包括：

- 一个包含所有调试工具的图形界面
  - 模拟器
  - 编程器 （单独销售）
  - 仿真器 （单独销售）
  - 在线调试器 （单独销售）
- 具有彩色上下文代码显示的全功能编辑器
- 多项目管理器
- 内容可直接编辑的可定制式数据窗口
- 高级源代码调试
- 可视化器件初始化程序，便于进行寄存器的初始化
- 鼠标停留在变量上进行查看的功能
- 通过拖放把变量从源代码窗口拉到观察窗口
- 丰富的在线帮助
- 集成了可选的第三方工具，如 HI-TECH 软件 C 编译器和 IAR C 编译器

MPLAB IDE 可以让您：

- 编辑源文件 （汇编语言或 C 语言）
- 点击一次即可完成汇编 （或编译）并将代码下载到 PICmicro MCU 仿真器和模拟器工具中 （自动更新所有项目信息）
- 可使用如下各项进行调试：
  - 源文件 （汇编语言或 C 语言）
  - 混合汇编语言和 C 语言
  - 机器码

MPLAB IDE 在单个开发范例中支持使用多种调试工具，包括从成本效益高的模拟器到低成本的在线调试器，再到全功能的仿真器。这样缩短了用户升级到更加灵活而功能更强大的工具时的学习时间。

25.2 MPASM 汇编器

MPASM 汇编器是全功能通用宏汇编器，适用于所有的 PICmicro MCU。

MPASM 汇编器可生成用于 MPLINK 目标链接器的可重定位目标文件、Intel® 标准 HEX 文件、详细描述存储器使用状况和符号参考的 MAP 文件、包含源代码行及生成机器码的绝对 LST 文件以及用于调试的 COFF 文件。

MPASM 汇编器具有如下特征：

- 集成在 MPLAB IDE 项目中
- 用户定义的宏可简化汇编代码
- 对多用途源文件进行条件汇编
- 允许完全控制汇编过程的指令

25.3 MPLAB C18 和 MPLAB C30 C 编译器

MPLAB C18 和 MPLAB C30 代码开发系统是完全的 ANSI C 编译器，分别适用于 Microchip 的 PIC18 系列单片机和 dsPIC30F 系列数据信号控制器。这些编译器可提供其他编译器并不具备的强大的集成功能和出众的代码优化能力，且使用方便。

为便于源代码调试，编译器提供了针对 MPLAB IDE 调试器的优化符号信息。

25.4 MPLINK 目标链接器 / MPLIB 目标库管理器

MPLINK 目标链接器包含了由 MPASM 汇编器、MPLAB C18 C 编译器产生的可重定位目标。通过使用链接器脚本中的指令，它还可链接预编译库中的可重定位目标。

MPLIB目标库管理器管理预编译代码库文件的创建和修改。当从源文件调用库中的一段子程序时，只有包含此子程序的模块被链接到应用中。这样可使大型库在许多不同应用中被高效地利用。

目标链接器 / 库管理器具有如下特征：

- 高效地连接单个的库而不是许多小文件
- 通过将相关的模块组合在一起来增强代码的可维护性
- 只要列出、替换、删除和抽取模块，便可灵活地创建库

25.5 MPLAB ASM30 汇编器、链接器和库管理器

MPLAB ASM30 汇编器为 dsPIC30F 器件提供转换自符号汇编语言的可重定位机器码。 MPLAB C30 C 编译器使用该汇编器生成目标文件。汇编器产生可重定位目标文件之后，可将这些目标文件存档，或与其他可重定位目标文件和存档链接以生成可执行文件。该汇编器有如下显著特征：

- 支持整个 dsPIC30F 指令集
- 支持定点数据和浮点数据
- 命令行界面
- 丰富的指令集
- 灵活的宏语言
- MPLAB IDE 兼容性

25.6 MPLAB SIM 软件模拟器

MPLAB SIM 软件模拟器在指令级对 PICmicro MCU 和 dsPIC® DSC 进行模拟，使得用户可以在 PC 主机的环境下进行代码开发。对于任何给定的指令，用户均可对数据区进行检查或修改，并通过各种触发机制来产生激励。可以将各寄存器的情况记录在文件中，以便进行进一步地运行时分析。跟踪缓冲器和逻辑分析器的显示使模拟器还能记录和跟踪程序的执行、 I/O 的动作以及内部寄存器的状况。

MPLAB SIM 软件模拟器完全支持使用 MPLAB C18 和 MPLAB C30 C 编译器以及MPASM和MAPLAB ASM30 汇编器的符号调试。该软件模拟器可用于在实验室环境外灵活地开发和调试代码，是一款完美且经济的软件开发工具。

25.7 MPLAB ICE 2000 高性能在线仿真器

MPLAB ICE 2000 在线仿真器旨在为产品开发工程师提供一整套用于 PICmicro 单片机的设计工具。 MPLAB ICE 2000 在线仿真器的软件控制由 MPLAB 集成开发环境平台提供，它允许在单一环境下进行编辑、编译、下载以及源代码调试。

MPLAB ICE 2000 是全功能仿真器系统，它具有增强的跟踪、触发和数据监控功能。处理器模块可插拔，使系统可轻松进行重新配置以适应各种不同处理器的仿真需要。MPLAB ICE 2000 在线仿真器的架构允许对其进行扩展以支持新的 PICmicro 单片机。

MPLAB ICE 2000 在线仿真器系统设计为一款实时仿真系统，该仿真系统具备通常只有昂贵的开发工具中才有的高级功能。选择 PC 平台和 Microsoft® Windows® 32 位操作系统可使这些功能在一个简单而统一的应用中得到很好的利用。

25.8 MPLAB ICE 4000 高性能在线仿真器

MPLAB ICE 4000 在线仿真器旨在为产品开发工程师提供一整套用于高端 PICmicro MCU 和 dsPIC DSC 的设计工具。 MPLAB ICE 4000 在线仿真器的软件控制由 MPLAB 集成开发环境平台提供，它允许在单一环境下进行编辑、编译、下载以及源代码调试。

MPLAB ICE 4000 是高级的仿真系统，除具备 MPLAB ICE 2000 的所有功能外，它还增加了适用于 dsPIC30F 和 PIC18XXXX 器件的仿真存储容量以及高速性能。该仿真器的先进特性包括复杂触发和定时功能及高达 2 Mb 的仿真存储容量。

MPLAB ICE 4000 在线仿真系统设计为一款实时仿真系统，该仿真系统具备通常只有在更加昂贵的开发工具中才有的高级功能。选择 PC 平台和 Microsoft Windows 32 位操作系统可使这些功能在一个简单而统一的应用程序中得以很好的利用。

25.9 MPLAB ICD 2 在线调试器

Microchip 的在线调试器 MPLAB ICD 2 是一款功能强大而成本低廉的运行时开发工具，通过 RS-232 或高速 USB 接口与 PC 主机相连。该工具基于闪存 PICmicro MCU，可用于开发本系列及其他 PICmicro MCU 和 dsPIC DSC。 MPLAB ICD 2 使用了闪存器件中内建的在线调试功能。该功能结合 Microchip 的在线串行编程（In-Circuit Serial Programming™， ICSP™）协议，可在 MPLAB 集成开发环境的图形用户界面上提供成本效益很高的在线闪存调试。这使设计人员可通过设置断点、单步运行以及对变量、CPU 状态以及外设寄存器进行监视的方法实现源代码的开发和调试。其全速运行特性可对硬件和应用进行实时测试。MPLAB ICD 2 还可用作某些 PICmicro 器件的开发编程器。

25.10 MPLAB PM3 器件编程器

MPLAB PM3 器件编程器是一款通用的、符合 CE 规范的器件编程器，其可编程电压设置在 VDDMIN 和 VDDMAX 之间时可靠性最高。它有一个用来显示菜单和错误信息的大 LCD 显示器（128 x 64），以及一个支持各种封装类型的可拆卸模块化插槽装置。编程器标准配置中带有一根 ICSP™ 电缆。在单机模式下， MPLAB PM3 器件编程器不必与 PC 相连即可对 PICmicro 器件进行读取、验证和编程。在该模式下它还可设置代码保护。MPLAB PM3 通过 RS-232 或 USB 电缆连接到 PC 主机上。MPLAB PM3 具备高速通信能力以及优化算法，可对存储器很大的器件进行快速编程，它还采用 SD/MMC 卡用作文件存储及数据安全应用。

25.11 PICSTART Plus 开发编程器

PICSTART Plus开发编程器是一款易于使用而成本低廉的原型编程器。它通过 COM（RS-232）端口与 PC 相连。MPLAB 集成开发环境软件使得该编程器的使用简便、高效。PICSTART Plus 开发编程器支持采用 DIP 封装的大部分 PICmicro 器件，其引脚数最多可达 40 个。引脚数更多的器件，如 PIC16C92X 和 PIC17C76X，可通过连接一个转接插槽来获得支持。PICSTART Plus 开发编程器符合 CE 规范。

25.12 演示、开发和评估板

有许多演示、开发和评估板可用于各种 PICmicro MCU 和 dsPIC DSC，实现对全功能系统的快速应用开发。大多数的演示、开发和评估板都有实验布线区，供用户添加定制电路；还有应用固件和源代码，用于测试和修改。

这些板支持多种功能部件，包括 LED、温度传感器、开关、扬声器、RS-232 接口、LCD 显示器、电位计和附加 EEPROM 存储器。

演示和开发板可用于教学环境，在实验布线区设计定制电路，从而掌握各种单片机应用。

除了 PICDEM™ 和 dsPICDEM™ 演示 / 开发板系列电路外，Microchip 还有一系列评估工具包和演示软件，适用于模拟滤波器设计、KEELOQ® 数据安全产品 IC、CAN、IrDA®、PowerSmart® 电池管理、SEEVAL® 评估系统、Σ–Δ ADC、流速传感器，等等。

有关演示、开发和评估工具包的完整列表，请查阅 Microchip 公司网页（www.microchip.com）以及最新的 *“Product Selector Guide（产品选型指南）”*（DS00148）。


注：

注：

26.0 电气规范

绝对最大值（†）

偏置电压下的环境温度 .... -40°C 至 +125°C
储存温度 .... -65°C 至 +150°C
任一引脚（除 V_{DD} 和 $\overline{MCLR}$ 外）相对于 V_{SS} 的电压 .... -0.3V 至（V_{DD} + 0.3V）
V_{DD} 相对于 V_{SS} 的电压 .... -0.3V 至 +7.5V
$\overline{MCLR}$ 引脚相对于 V_{SS} 的电压（注 **2**） .... 0V 至 +13.25V
总功耗（注 **1**） 1.0W
V_{SS} 引脚最大输出电流 .... 300 mA
V_{DD} 引脚最大输入电流 .... 250 mA
输入箝位电流 I_{IK}（$V_I < 0$ 或 $V_I > V_{DD}$） .... ± 20 mA
输出箝位电流 I_{OK}（$V_O < 0$ 或 $V_O > V_{DD}$） .... ± 20 mA
任一 I/O 引脚的最大灌电流 .... 25 mA
任一 I/O 引脚的最大拉电流 .... 25 mA
所有端口的最大灌电流 .... 200 mA
所有端口的最大拉电流 .... 200 mA

注 **1**：功耗按如下公式计算：
$P_{dis} = V_{DD} \times \{I_{DD} - \sum I_{OH}\} + \sum \{(V_{DD} - V_{OH}) \times I_{OH}\} + \sum(V_{OL} \times I_{OL})$

2：如果 $\overline{MCLR}$/V_{PP}/RE3 引脚上的尖峰电压低于 V_{SS}，感应电流大于 80mA，可能会引起器件锁死。因此当 $\overline{MCLR}$/V_{PP}/RE3 引脚驱动为低电平时，应该串联一个 50-100Ω 的电阻，而不是直接把该引脚连接到 V_{SS}。

† 注意：如果器件工作条件超过上述 “绝对最大值”，可能会对器件造成永久性损坏。上述值仅为运行条件极大值，我们建议不要使器件在该规范规定的范围以外运行。器件长时间工作在最大值条件下，其稳定性会受到影响。

图 26-1： **PIC18F6390/6490/8390/8490** 电压－频率关系图 （工业级）

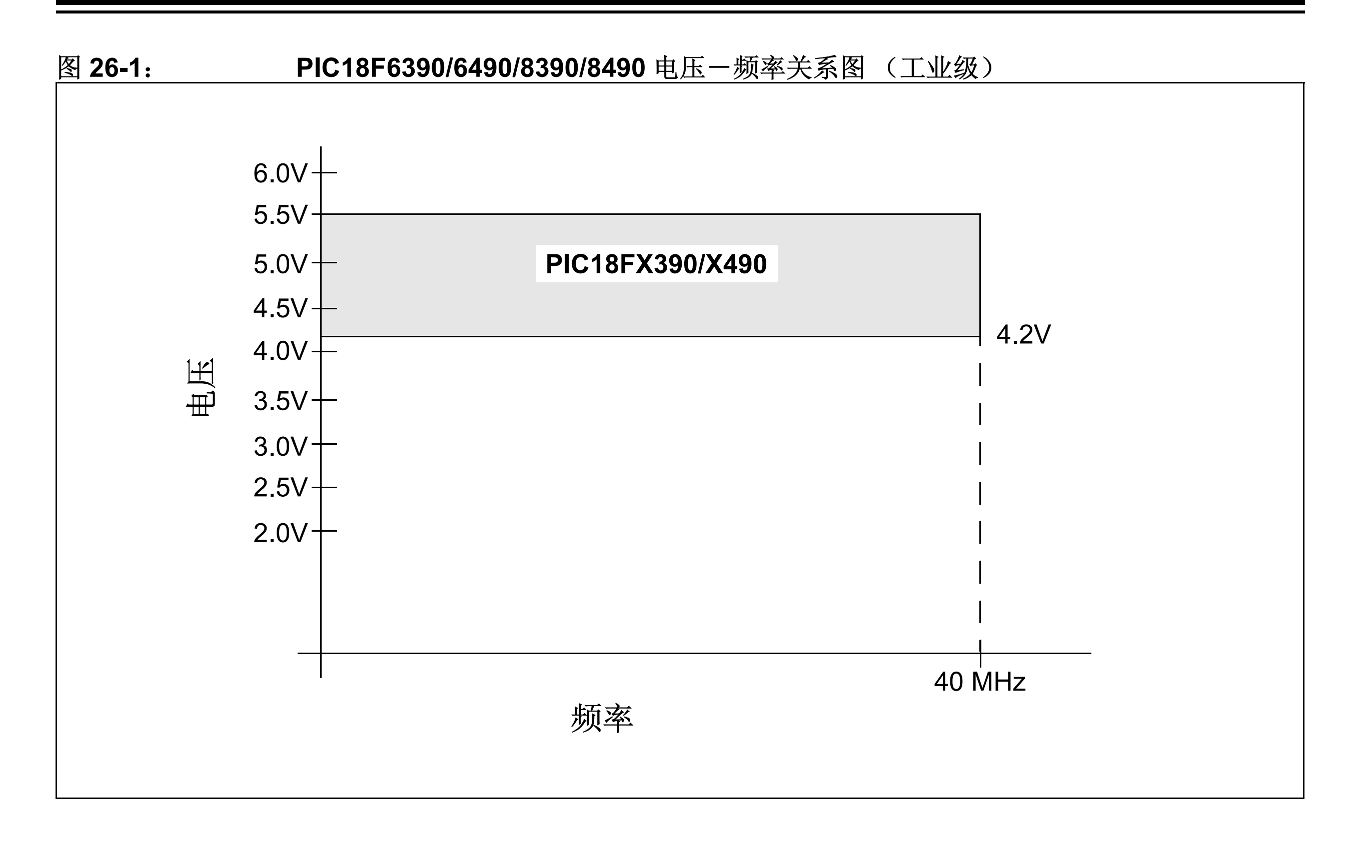
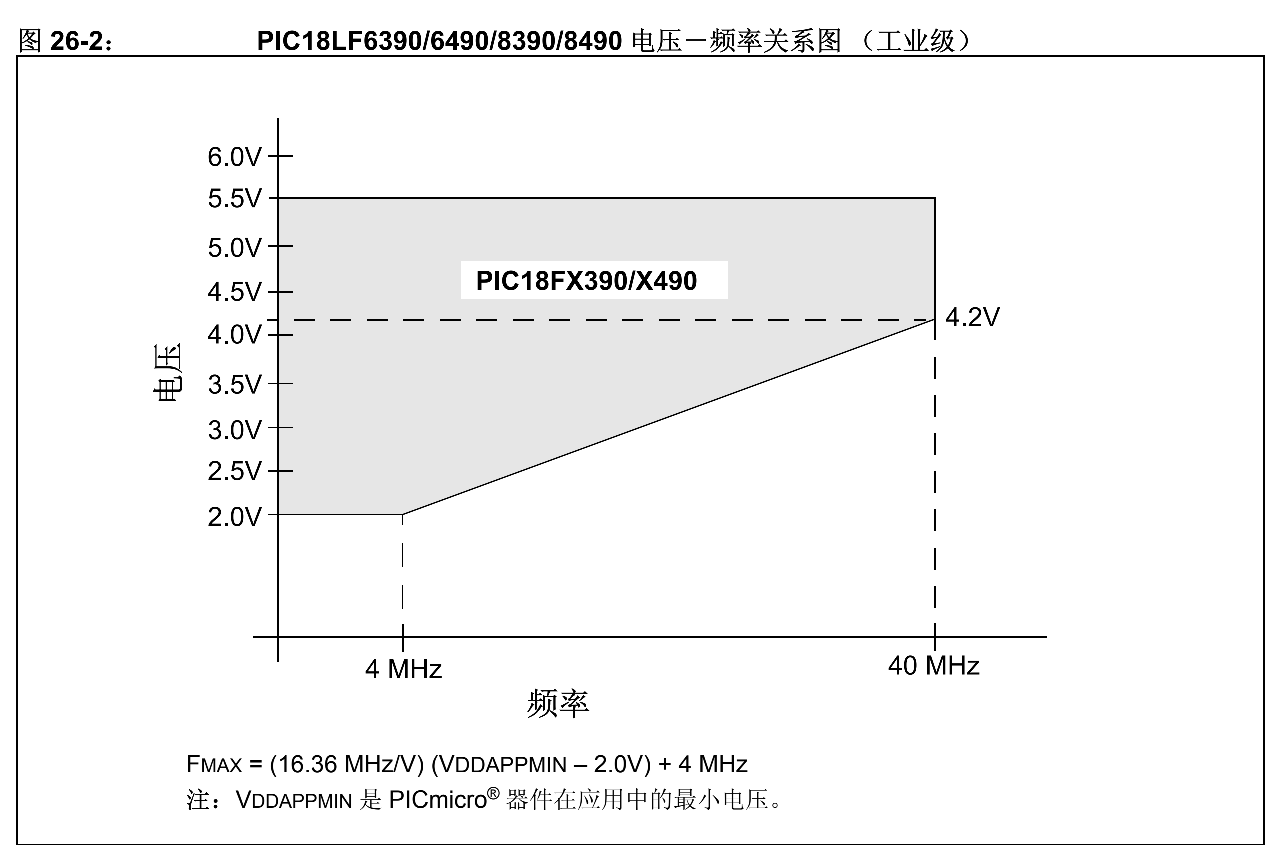


图 26-2： **PIC18LF6390/6490/8390/8490** 电压－频率关系图 （工业级）



FMAX = (16.36 MHz/V) (VDDAPPMIN – 2.0V) + 4 MHz

注：VDDAPPMIN 是 PICmicro® 器件在应用中的最小电压。

26.1 直流规范： 供电电压
PIC18F6390/6490/8390/8490 （工业级）
PIC18LF6390/6490/8390/8490 （工业级）

PIC18LF6390/6490/8390/8490 （工业级）			标准工作条件 （除非另行声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级）				
PIC18F6390/6490/8390/8490 （工业级）			标准工作条件 （除非另行声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级）				
参数编号	符号	特性	最小值	典型值	最大值	单位	条件
D001	VDD	供电电压					
		PIC18LFX390/X490	2.0	—	5.5	V	HS、 XT、 RC 和 LP 振荡器模式
		PIC18FX390/X490	4.2	—	5.5	V	
D002	VDR	**RAM** 数据保存电压 [1]	1.5	—	—	V	
D003	VPOR	**VDD** 启动电压 （确保内部上电复位信号）	—	—	0.7	V	详情请参见关于上电复位的章节
D004	SVDD	**VDD** 上升率 （确保内部上电复位信号）	0.05	—	—	V/ms	详情请参见关于上电复位的章节
	VBOR	欠压复位电压					
D005		PIC18LFX390/X490					
		BORV1:BORV0 = 11	2.00	2.05	2.16	V	
		BORV1:BORV0 = 10	2.65	2.79	2.93	V	
D005		所有器件					
		BORV1:BORV0 = 01	4.11	4.33	4.55	V	
		BORV1:BORV0 = 00	4.36	4.59	4.82	V	

图注： 阴影行是为了增强表的可读性。

注 **1：** 该电压是休眠模式或器件复位状态下在不丢失 RAM 数据的前提下的最小 VDD。

26.2 直流规范： 掉电和供电电流 PIC18F6390/6490/8390/8490 （工业级） PIC18LF6390/6490/8390/8490 （工业级）

PIC18LF6390/6490/8390/8490 （工业级）		**标准工作条件 （除非另行声明）** 工作温度 -40°C ≤ TA ≤ +85°C （工业级）				
PIC18F6390/6490/8390/8490 （工业级）		**标准工作条件 （除非另行声明）** 工作温度 -40°C ≤ TA ≤ +85°C （工业级）				
参数编号	**器件**	**典型值**	**最大值**	**单位**	**条件**	
	掉电电流 （IPD）[1]					
	PIC18LFX390/X490	0.1	0.5	μA	-40°C	VDD = 2.0V，（休眠模式）
		0.1	0.5	μA	+25°C	
		0.2	2.0	μA	+85°C	
	PIC18LFX390/X490	0.1	0.5	μA	-40°C	VDD = 3.0V，（休眠模式）
		0.1	0.5	μA	+25°C	
		0.3	5	μA	+85°C	
	所有器件	0.1	2.0	μA	-40°C	VDD = 5.0V，（休眠模式）
		0.1	2.0	μA	+25°C	
		0.4	10	μA	+85°C	

图注： TBD = 待定。阴影行是为了增强表的可读性。

注 1： 休眠模式下的掉电电流不是由振荡器类型决定的。掉电电流是在器件休眠时，所有 I/O 引脚处于高阻态并且连接到 VDD 或者 VSS，禁止所有会带来新增电流的功能部件 （比如 WDT、 Timer1 振荡器和 BOR 等）时测得的。

2： 供电电流主要是由工作电压、频率和模式一起决定的。其他因素，如 I/O 引脚负载和开关频率、振荡器类型和电路、内部代码执行模式和温度也会影响电流消耗。
在正常工作模式下，所有 IDD 测量的测试条件为：
OSC1 = 外部方波，满幅；所有 I/O 引脚均为三态，拉至 VDD 或 VSS；
MCLR = VDD；根据具体应用使能或禁止 WDT。

3： 选择低功耗 Timer1 振荡器。

4： BOR 和 HLVD 使能内部带隙参考源。当同时使能两个模块时，电流消耗将少于两个规范值的和。

26.2 直流规范： 掉电和供电电流 PIC18F6390/6490/8390/8490 （工业级） PIC18LF6390/6490/8390/8490 （工业级） （续）

PIC18LF6390/6490/8390/8490 （工业级）		标准工作条件 （除非另行声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级）					
PIC18F6390/6490/8390/8490 （工业级）		**标准工作条件 （除非另行声明）** 工作温度 -40°C ≤ TA ≤ +85°C （工业级）					
参数编号	**器件**	**典型值**	**最大值**	**单位**	**条件**		
	供电电流 （IDD）[(2)]						
	PIC18LFX390/X490	12	26	μA	-40°C	VDD = 2.0V	FOSC = 31 kHz （**RC_RUN** 模式， INTRC 时钟源）
		12	24	μA	+25°C		
		12	23	μA	+85°C		
	PIC18LFX390/X490	32	50	μA	-40°C	VDD = 3.0V	
		27	48	μA	+25°C		
		22	46	μA	+85°C		
	所有器件	84	134	μA	-40°C	VDD = 5.0V	
		82	128	μA	+25°C		
		72	122	μA	+85°C		
	PIC18LFX390/X490	.26	.8	mA	-40°C	VDD = 2.0V	FOSC = 1 MHz （**RC_RUN** 模式， INTOSC 时钟源）
		.26	.8	mA	+25°C		
		.26	.8	mA	+85°C		
	PIC18LFX390/X490	.48	1.04	mA	-40°C	VDD = 3.0V	
		.44	.96	mA	+25°C		
		.48	.88	mA	+85°C		
	所有器件	.88	1.84	mA	-40°C	VDD = 5.0V	
		.88	1.76	mA	+25°C		
		.8	1.68	mA	+85°C		

图注： TBD = 待定。阴影行是为了增强表的可读性。

注 **1：** 休眠模式下的掉电电流不是由振荡器类型决定的。掉电电流是在器件休眠时，所有 I/O 引脚处于高阻态并且连接到 VDD 或者 VSS，禁止所有会带来新增电流的功能部件 （比如 WDT、 Timer1 振荡器和 BOR 等）时测得的。

2： 供电电流主要是由工作电压、频率和模式一起决定的。其他因素，如 I/O 引脚负载和开关频率、振荡器类型和电路、内部代码执行模式和温度也会影响电流消耗。
在正常工作模式下，所有 IDD 测量的测试条件为：
OSC1 = 外部方波，满幅；所有 I/O 引脚均为三态，拉至 VDD 或 VSS ；
MCLR = VDD ；根据具体应用使能或禁止 WDT。

3： 选择低功耗 Timer1 振荡器。

4： BOR 和 HLVD 使能内部带隙参考源。当同时使能两个模块时，电流消耗将少于两个规范值的和。

26.2 直流规范： 掉电和供电电流 PIC18F6390/6490/8390/8490 （工业级） PIC18LF6390/6490/8390/8490 （工业级） （续）

PIC18LF6390/6490/8390/8490 （工业级）		标准工作条件 （除非另行声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级）					
PIC18F6390/6490/8390/8490 （工业级）		标准工作条件 （除非另行声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级）					
参数编号	器件	典型值	最大值	单位	条件		
	供电电流 （IDD）[2]						
	PIC18LFX390/X490	0.6	1.7	μA	-40°C	VDD = 2.0V	FOSC = 4 MHz （**RC_RUN** 模式， INTOSC 时钟源）
		0.6	1.6	μA	+25°C		
		0.6	1.5	μA	+85°C		
	PIC18LFX390/X490	1.0	2.4	mA	-40°C	VDD = 3.0V	
		1.0	2.4	mA	+25°C		
		1.0	2.4	mA	+85°C		
	所有器件	2.0	4.2	mA	-40°C	VDD = 5.0V	
		2.0	4	mA	+25°C		
		2.0	3.8	mA	+85°C		
	PIC18LFX390/X490	2.3	6.4	μA	-40°C	VDD = 2.0V	FOSC = 31 kHz （**RC_IDLE** 模式， INTRC 时钟源）
		2.5	6.4	μA	+25°C		
		2.9	8.8	μA	+85°C		
	PIC18LFX390/X490	3.6	8.8	μA	-40°C	VDD = 3.0V	
		3.8	8.8	μA	+25°C		
		4.6	12	μA	+85°C		
	所有器件	7.4	13	μA	-40°C	VDD = 5.0V	
		7.8	13	μA	+25°C		
		9.1	29	μA	+85°C		

图注： TBD = 待定。阴影行是为了增强表的可读性。

注 **1：** 休眠模式下的掉电电流不是由振荡器类型决定的。掉电电流是在器件休眠时，所有 I/O 引脚处于高阻态并且连接到 VDD 或者 VSS，禁止所有会带来新增电流的功能部件 （比如 WDT、 Timer1 振荡器和 BOR 等）时测得的。

2： 供电电流主要是由工作电压、频率和模式一起决定的。其他因素，如 I/O 引脚负载和开关频率、振荡器类型和电路、内部代码执行模式和温度也会影响电流消耗。
在正常工作模式下，所有 IDD 测量的测试条件为：
OSC1 = 外部方波，满幅；所有 I/O 引脚均为三态，拉至 VDD 或 VSS ；
MCLR = VDD ；根据具体应用使能或禁止 WDT。

3： 选择低功耗 Timer1 振荡器。

4： BOR 和 HLVD 使能内部带隙参考源。当同时使能两个模块时，电流消耗将少于两个规范值的和。

26.2 直流规范： 掉电和供电电流 PIC18F6390/6490/8390/8490 （工业级） PIC18LF6390/6490/8390/8490 （工业级） （续）

PIC18LF6390/6490/8390/8490 （工业级）		标准工作条件 （除非另行声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级）					
PIC18F6390/6490/8390/8490 （工业级）		标准工作条件 （除非另行声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级）					
参数编号	器件	典型值	最大值	单位	条件		
	供电电流 （IDD）[2]						
	PIC18LFX390/X490	132	280	μA	-40°C	VDD = 2.0V	FOSC = 1 MHz （**RC_IDLE** 模式， INTOSC 时钟源）
		140	280	μA	+25°C		
		152	280	μA	+85°C		
	PIC18LFX390/X490	200	400	μA	-40°C	VDD = 3.0V	
		216	400	μA	+25°C		
		252	400	μA	+85°C		
	所有器件	.40	.8	mA	-40°C	VDD = 5.0V	
		.42	.8	mA	+25°C		
		.44	.8	mA	+85°C		
	PIC18LFX390/X490	272	400	μA	-40°C	VDD = 2.0V	FOSC = 4 MHz （**RC_IDLE** 模式， INTOSC 时钟源）
		280	400	μA	+25°C		
		288	400	μA	+85°C		
	PIC18LFX390/X490	416	720	μA	-40°C	VDD = 3.0V	
		432	720	μA	+25°C		
		464	720	μA	+85°C		
	所有器件	.8	1.3	mA	-40°C	VDD = 5.0V	
		.9	1.2	mA	+25°C		
		.9	1.1	mA	+85°C		

图注： TBD = 待定。阴影行是为了增强表的可读性。

注 **1：** 休眠模式下的掉电电流不是由振荡器类型决定的。掉电电流是在器件休眠时，所有 I/O 引脚处于高阻态并且连接到 VDD 或者 VSS，禁止所有会带来新增电流的功能部件 （比如 WDT、 Timer1 振荡器和 BOR 等）时测得的。

2： 供电电流主要是由工作电压、频率和模式一起决定的。其他因素，如 I/O 引脚负载和开关频率、振荡器类型和电路、内部代码执行模式和温度也会影响电流消耗。

在正常工作模式下，所有 IDD 测量的测试条件为：

OSC1 = 外部方波，满幅；所有 I/O 引脚均为三态，拉至 VDD 或 VSS ；

MCLR = VDD ；根据具体应用使能或禁止 WDT。

3： 选择低功耗 Timer1 振荡器。

4： BOR 和 HLVD 使能内部带隙参考源。当同时使能两个模块时，电流消耗将少于两个规范值的和。

26.2 直流规范： 掉电和供电电流 PIC18F6390/6490/8390/8490 （工业级） PIC18LF6390/6490/8390/8490 （工业级） （续）

PIC18LF6390/6490/8390/8490 （工业级）	标准工作条件 （除非另行声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级）					
PIC18F6390/6490/8390/8490 （工业级）	**标准工作条件 （除非另行声明）** 工作温度 -40°C ≤ TA ≤ +85°C （工业级）					
参数编号	**器件**	**典型值**	**最大值**	**单位**	**条件**		

参数编号	器件	典型值	最大值	单位	条件		
	供电电流 （IDD）[2]						
	PIC18LFX390/X490	250	500	µA	-40°C	VDD = 2.0V	FOSC = 1 MHz（**PRI_RUN** 模式，EC 振荡器）
		260	500	µA	+25°C		
		250	500	µA	+85°C		
	PIC18LFX390/X490	550	650	µA	-40°C	VDD = 3.0V	
		480	650	µA	+25°C		
		460	650	µA	+85°C		
	所有器件	1.2	1.6	mA	-40°C	VDD = 5.0V	
		1.1	1.5	mA	+25°C		
		1.0	1.4	mA	+85°C		
	PIC18LFX390/X490	0.72	2.0	mA	-40°C	VDD = 2.0V	FOSC = 4 MHz（**PRI_RUN** 模式，EC 振荡器）
		0.74	2.0	mA	+25°C		
		0.74	2.0	mA	+85°C		
	PIC18LFX390/X490	1.3	3.0	mA	-40°C	VDD = 3.0V	
		1.3	3.0	mA	+25°C		
		1.3	3.0	mA	+85°C		
	所有器件	2.7	6.0	mA	-40°C	VDD = 5.0V	
		2.6	6.0	mA	+25°C		
		2.5	6.0	mA	+85°C		
	所有器件	15	35	mA	-40°C	VDD = 4.2V	FOSC = 40 MHz（**PRI_RUN** 模式，EC 振荡器）
		16	35	mA	+25°C		
		16	35	mA	+85°C		
	所有器件	21	40	mA	-40°C	VDD = 5.0V	
		21	40	mA	+25°C		
		21	40	mA	+85°C		

图注： TBD = 待定。阴影行是为了增强表的可读性。

注 1： 休眠模式下的掉电电流不是由振荡器类型决定的。掉电电流是在器件休眠时，所有 I/O 引脚处于高阻态并且连接到 VDD 或者 VSS，禁止所有会带来新增电流的功能部件 （比如 WDT、 Timer1 振荡器和 BOR 等）时测得的。

2： 供电电流主要是由工作电压、频率和模式一起决定的。其他因素，如 I/O 引脚负载和开关频率、振荡器类型和电路、内部代码执行模式和温度也会影响电流消耗。
在正常工作模式下，所有 IDD 测量的测试条件为：
OSC1 = 外部方波，满幅；所有 I/O 引脚均为三态，拉至 VDD 或 VSS ；
MCLR = VDD ；根据具体应用使能或禁止 WDT。

3： 选择低功耗 Timer1 振荡器。

4： BOR 和 HLVD 使能内部带隙参考源。当同时使能两个模块时，电流消耗将少于两个规范值的和。

26.2 直流规范： 掉电和供电电流 PIC18F6390/6490/8390/8490 （工业级） PIC18LF6390/6490/8390/8490 （工业级） （续）

PIC18LF6390/6490/8390/8490（工业级）		标准工作条件 （除非另行声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级）					
PIC18F6390/6490/8390/8490（工业级）		标准工作条件 （除非另行声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级）					
参数编号	器件	典型值	最大值	单位	条件		
	供电电流 （IDD）[2]						
	所有器件	7.5	16	mA	-40°C	VDD = 4.2V	FOSC = 4 MHz. 16MHz 内部 (**PRI_RUN HS+PLL** 模式)
		7.4	15	mA	+25°C		
		7.3	14	mA	+85°C		
	所有器件	10	21	mA	-40°C	VDD = 5.0V	FOSC = 4 MHz， 16 MHz 内部 (**PRI_RUN HS+PLL** 模式)
		10	20	mA	+25°C		
		9.7	19	mA	+85°C		
	所有器件	17	35	mA	-40°C	VDD = 4.2V	FOSC = 10 MHz， 40MHz 内部 (**PRI_RUN HS+PLL** 模式)
		17	35	mA	+25°C		
		17	35	mA	+85°C		
	所有器件	23	40	mA	-40°C	VDD = 5.0V	FOSC = 10 MHz， 40MHz 内部 (**PRI_RUN HS+PLL** 模式)
		23	40	mA	+25°C		
		23	40	mA	+85°C		

图注： TBD = 待定。阴影行是为了增强表的可读性。

注 **1：** 休眠模式下的掉电电流不是由振荡器类型决定的。掉电电流是在器件休眠时，所有 I/O 引脚处于高阻态并且连接到 VDD 或者 VSS，禁止所有会带来新增电流的功能部件 （比如 WDT、 Timer1 振荡器和 BOR 等）时测得的。

2： 供电电流主要是由工作电压、频率和模式一起决定的。其他因素，如 I/O 引脚负载和开关频率、振荡器类型和电路、内部代码执行模式和温度也会影响电流消耗。
在正常工作模式下，所有 IDD 测量的测试条件为：
OSC1 = 外部方波，满幅；所有 I/O 引脚均为三态，拉至 VDD 或 VSS ；
MCLR = VDD ；根据具体应用使能或禁止 WDT。

3： 选择低功耗 Timer1 振荡器。

4： BOR 和 HLVD 使能内部带隙参考源。当同时使能两个模块时，电流消耗将少于两个规范值的和。

26.2 直流规范： 掉电和供电电流 PIC18F6390/6490/8390/8490 （工业级） PIC18LF6390/6490/8390/8490 （工业级）（续）

PIC18LF6390/6490/8390/8490 （工业级）		标准工作条件（除非另行声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级）					
PIC18F6390/6490/8390/8490 （工业级）		标准工作条件（除非另行声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级）					
参数编号	器件	典型值	最大值	单位	条件		
	供电电流（IDD）[2]						
	PIC18LFX390/X490	59	117	μA	-40°C	VDD = 2.0V	FOSC = 1 MHz（**PRI_IDLE** 模式，EC 振荡器）
		59	108	μA	+25°C		
		63	104	μA	+85°C		
	PIC18LFX390/X490	108	243	μA	-40°C	VDD = 3.0V	
		108	225	μA	+25°C		
		117	216	μA	+85°C		
	所有器件	270	432	μA	-40°C	VDD = 5.0V	
		216	405	μA	+25°C		
		270	387	μA	+85°C		
	PIC18LFX390/X490	234	428	μA	-40°C	VDD = 2.0V	FOSC = 4 MHz（**PRI_IDLE** 模式，EC 振荡器）
		230	405	μA	+25°C		
		243	387	μA	+85°C		
	PIC18LFX390/X490	378	810	μA	-40°C	VDD = 3.0V	
		387	765	μA	+25°C		
		405	729	μA	+85°C		
	所有器件	.8	1.35	mA	-40°C	VDD = 5.0V	
		.8	1.26	mA	+25°C		
		.8	1.17	mA	+85°C		
	所有器件	5.4	14.4	mA	-40°C	VDD = 4.2V	FOSC = 40 MHz（**PRI_IDLE** 模式，EC 振荡器）
		5.6	14.4	mA	+25°C		
		5.9	14.4	mA	+85°C		
	所有器件	7.3	16.2	mA	-40°C	VDD = 5.0V	
		8.2	16.2	mA	+25°C		
		7.5	16.2	mA	+85°C		

图注： TBD = 待定。阴影行是为了增强表的可读性。

注 **1：** 休眠模式下的掉电电流不是由振荡器类型决定的。掉电电流是在器件休眠时，所有 I/O 引脚处于高阻态并且连接到 VDD 或者 VSS，禁止所有会带来新增电流的功能部件（比如 WDT、 Timer1 振荡器和 BOR 等）时测得的。

2： 供电电流主要是由工作电压、频率和模式一起决定的。其他因素，如 I/O 引脚负载和开关频率、振荡器类型和电路、内部代码执行模式和温度也会影响电流消耗。
在正常工作模式下，所有 IDD 测量的测试条件为：
OSC1 = 外部方波，满幅；所有 I/O 引脚均为三态，拉至 VDD 或 VSS ；
MCLR = VDD ；根据具体应用使能或禁止 WDT。

3： 选择低功耗 Timer1 振荡器。

4： BOR 和 HLVD 使能内部带隙参考源。当同时使能两个模块时，电流消耗将少于两个规范值的和。

 初稿

26.2 直流规范： 掉电和供电电流 PIC18F6390/6490/8390/8490 （工业级） PIC18LF6390/6490/8390/8490 （工业级） （续）

PIC18LF6390/6490/8390/8490 （工业级）		标准工作条件 （除非另行声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级）					
PIC18F6390/6490/8390/8490 （工业级）		标准工作条件 （除非另行声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级）					
参数编号	**器件**	**典型值**	**最大值**	**单位**	**条件**		
	供电电流 （IDD）[2]						
	PIC18LFX390/X490	13	9	μA	-10°C	VDD = 2.0V	FOSC = 32 kHz[4]（**SEC_RUN** 模式，Timer1 作为时钟源）
		14	9	μA	+25°C		
		16	11	μA	+70°C		
	PIC18LFX390/X490	34	12	μA	-10°C	VDD = 3.0V	
		31	12	μA	+25°C		
		28	14	μA	+70°C		
	所有器件	72	20	μA	-10°C	VDD = 5.0V	
		65	20	μA	+25°C		
		59	25	μA	+70°C		
	PIC18LFX390/X490	5.5	15	μA	-10°C	VDD = 2.0V	FOSC = 32 kHz[4]（**SEC_IDLE** 模式，Timer1 作为时钟源）
		5.8	15	μA	+25°C		
		6.1	18	μA	+70°C		
	PIC18LFX390/X490	8.2	30	μA	-10°C	VDD = 3.0V	
		8.6	30	μA	+25°C		
		8.8	35	μA	+70°C		
	所有器件	13	80	μA	-10°C	VDD = 5.0V	
		13	80	μA	+25°C		
		13	85	μA	+70°C		

图注： TBD = 待定。阴影行是为了增强表的可读性。

注 **1：** 休眠模式下的掉电电流不是由振荡器类型决定的。掉电电流是在器件休眠时，所有 I/O 引脚处于高阻态并且连接到 VDD 或者 VSS，禁止所有会带来新增电流的功能部件 （比如 WDT、 Timer1 振荡器和 BOR 等）时测得的。

2： 供电电流主要是由工作电压、频率和模式一起决定的。其他因素，如 I/O 引脚负载和开关频率、振荡器类型和电路、内部代码执行模式和温度也会影响电流消耗。
在正常工作模式下，所有 IDD 测量的测试条件为：
OSC1 = 外部方波，满幅；所有 I/O 引脚均为三态，拉至 VDD 或 VSS ；
MCLR = VDD ；根据具体应用使能或禁止 WDT。

3： 选择低功耗 Timer1 振荡器。

4： BOR 和 HLVD 使能内部带隙参考源。当同时使能两个模块时，电流消耗将少于两个规范值的和。

26.2 直流规范： 掉电和供电电流 PIC18F6390/6490/8390/8490 （工业级） PIC18LF6390/6490/8390/8490 （工业级）（续）

PIC18LF6390/6490/8390/8490 （工业级）		标准工作条件 （除非另行声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级）					
PIC18F6390/6490/8390/8490 （工业级）		标准工作条件 （除非另行声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级）					
参数编号	器件	典型值	最大值	单位	条件		
	模块差分电流 （ΔIWDT、ΔIBOR、ΔILVD、ΔLCD、ΔIOSCB 和 ΔIAD）						
D022 （ΔIWDT）	看门狗定时器	1.7	4.0	μA	-40°C	VDD = 2.0V	
		2.1	4.0	μA	+25°C		
		2.6	5.0	μA	+85°C		
		2.2	6.0	μA	-40°C	VDD = 3.0V	
		2.4	6.0	μA	+25°C		
		2.8	7.0	μA	+85°C		
		2.9	10.0	μA	-40°C	VDD = 5.0V	
		3.1	10.0	μA	+25°C		
		3.3	13.0	μA	+85°C		
D022A (ΔIBOR)	欠压复位	17	35.0	μA	-40°C 至 +85°C	VDD = 3.0V	
		47	45.0	μA	-40°C 至 +85°C	VDD = 5.0V	
D022B (ΔILVD)	高 / 低压检测	14	25.0	μA	-40°C 至 +85°C	VDD = 2.0V	
		18	35.0	μA	-40°C 至 +85°C	VDD = 3.0V	
		21	45.0	μA	-40°C 至 +85°C	VDD = 5.0V	
D024 (ΔILVD)	**LCD 模块**	15	TBD	μA	-40°C 至 +85°C	VDD = 3.0V	
		20	TBD	μA	-40°C 至 +85°C	VDD = 5.0V	
D025 (ΔIOSCB)	**Timer1 振荡器**	1.0	3.5	μA	-10°C	VDD = 2.0V	Timer1(4) 为 32 kHz
		1.1	3.5	μA	+25°C		
		1.1	4.5	μA	+70°C		
		1.2	4.5	μA	-10°C	VDD = 3.0V	Timer1(4) 为 32 kHz
		1.3	4.5	μA	+25°C		
		1.2	5.5	μA	+70°C		
		1.8	6.0	μA	-10°C	VDD = 5.0V	Timer1(4) 为 32 kHz
		1.9	6.0	μA	+25°C		
		1.9	7.0	μA	+70°C		
D026 (ΔIAD)	**A/D 转换器**	1.0	3.0	μA	—	VDD = 2.0V	A/D 启动，但不进行转换
		1.0	4.0	μA	—	VDD = 3.0V	
		1.0	8.0	μA	—	VDD = 5.0V	

图注： TBD = 待定。阴影行是为了增强表的可读性。

注 1： 休眠模式下的掉电电流不是由振荡器类型决定的。掉电电流是在器件休眠时，所有 I/O 引脚处于高阻态并且连接到 VDD 或者 VSS，禁止所有会带来新增电流的功能部件 （比如 WDT、 Timer1 振荡器和 BOR 等）时测得的。

2： 供电电流主要是由工作电压、频率和模式一起决定的。其他因素，如 I/O 引脚负载和开关频率、振荡器类型和电路、内部代码执行模式和温度也会影响电流消耗。
在正常工作模式下，所有 IDD 测量的测试条件为：
OSC1 = 外部方波，满幅；所有 I/O 引脚均为三态，拉至 VDD 或 VSS ；
MCLR = VDD ；根据具体应用使能或禁止 WDT。

3： 选择低功耗 Timer1 振荡器。

4： BOR 和 HLVD 使能内部带隙参考源。当同时使能两个模块时，电流消耗将少于两个规范值的和。

26.3 直流规范： PIC18F6390/6490/8390/8490 （工业级） PIC18LF6390/6490/8390/8490 （工业级）

直流规范			标准工作条件 （除非另行声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级）			
参数编号	符号	特性	最小值	最大值	单位	条件
	VIL	输入低电压				
		I/O 端口：				
D030		带 TTL 缓冲器	VSS	0.15 VDD	V	VDD < 4.5V
D030A			—	0.8	V	4.5V ≤ Vdd ≤ 5.5V
D031		带施密特触发缓冲器	VSS	0.2 VDD	V	
		RC3 和 RC4	VSS	0.3 VDD	V	
D032		$\overline{\text{MCLR}}$	VSS	0.2 VDD	V	
D032A		OSC1 和 T1OSI	VSS	0.3 VDD	V	LP、 XT、 HS 和 HSPLL 模式 [(1)]
D033		OSC1	VSS	0.2 VDD	V	EC 模式 [(1)]
	VIH	输入高电压				
		I/O 端口：				
D040		带 TTL 缓冲器	0.25 VDD + 0.8V	VDD	V	VDD < 4.5V
D040A			2.0	VDD	V	4.5V ≤ Vdd ≤ 5.5V
D041		带施密特触发缓冲器	0.8 VDD	VDD	V	
		RC3 和 RC4	0.7 VDD	VDD	V	
D042		$\overline{\text{MCLR}}$	0.8 VDD	VDD	V	
D042A		OSC1 和 T1OSI	0.7 VDD	VDD	V	LP、 XT、 HS 和 HSPLL 模式 [(1)]
D043		OSC1	0.8 VDD	VDD	V	EC 模式 [(1)]
	IIL	输入泄漏电流 [（2,3）]				
D060		I/O 端口	—	±1	μA	VSS ≤ VPIN ≤ VDD，引脚处于高阻态
D061		$\overline{\text{MCLR}}$	—	±5	μA	VSS ≤ VPIN ≤ VDD
D063		OSC1	—	±5	μA	VSS ≤ VPIN ≤ VDD
	IPU	弱上拉电流				
D070	IPURB	PORTB 弱上拉电流	50	400	μA	VDD = 5V， VPIN = VSS

注 1： 在 RC 振荡器配置中， OSC1/CLKI 引脚被配置为施密特触发器输入。在 RC 模式下，建议不要使用外部时钟驱动 PICmicro® 器件。
2： $\overline{\text{MCLR}}$ 引脚上的泄漏电流主要取决于施加在该引脚上的电平。规定电平为正常工作条件下的电平。在不同的输入电压下可测得更高的泄漏电流。
3： 负电流定义为引脚的拉电流。
4： 参数仅为特征值，未经测试。

26.3 直流规范： PIC18F6390/6490/8390/8490 （工业级） PIC18LF6390/6490/8390/8490 （工业级）（续）

直流规范			标准工作条件 （除非另行声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级）			
参数编号	符号	特性	最小值	最大值	单位	条件
	VOL	输出低电压				
D080		I/O 端口	—	0.6	V	IOL = 8.5 mA， VDD = 4.5V， -40°C 至 +85°C
D083		OSC2/CLKO (RC、 RCIO、 EC 和 ECIO 模式)	—	0.6	V	IOL = 1.6 mA， VDD = 4.5V， -40°C 至 +85°C
	VOH	输出高电压 (3)				
D090		I/O 端口	VDD -0.7	—	V	IOH = -3.0 mA， VDD = 4.5V， -40°C 至 +85°C
D092		OSC2/CLKO (RC、 RCIO、 EC 和 ECIO 模式)	VDD -0.7	—	V	IOH = -1.3 mA， VDD = 4.5V， -40°C 至 +85°C
D150	VOD	漏级开路高电压	—	8.5	V	RA4 引脚
		输出引脚上的 容性负载规范				
D100(4)	COSC2	OSC2 引脚	—	15	pF	当外部时钟用于驱动 OSC1 时处于 XT、 HS 和 LP 模式
D101	CIO	所有 I/O 引脚和 OSC2 （在 RC 模式下）	—	50	pF	满足交流时序规范
D102	CB	SCL， SDA	—	400	pF	I2C™ 规范

注 1： 在 RC 振荡器配置中， OSC1/CLKI 引脚被配置为施密特触发器输入。在 RC 模式下，建议不要使用外部时钟驱动 PICmicro® 器件。
2： MCLR 引脚上的泄漏电流主要取决于施加在该引脚上的电平。规定电平为正常工作条件下的电平。在不同的输入电压下可测得更高的泄漏电流。
3： 负电流定义为引脚的拉电流。
4： 参数仅为特征值，未经测试。

表 **26-1**： 存储器编程要求

直流规范			标准工作条件（除非另行声明） 工作温度 -40°C ≤ TA ≤ +85°C（工业级）				
参数编号	符号	特性	最小值	典型值†	最大值	单位	条件
		闪存程序存储器					
D110	VPP	$\overline{MCLR}$/VPP 引脚上的电压	10.0	—	12.0	V	
D113	IDDP	编程期间的供电电流	—	—	1	mA	
D130	EP	耐擦写能力	—	1K	—	E/W	-40°C 至 +85°C
D131	VPR	用于读入的 VDD	VMIN	—	5.5	V	VMIN = 最小工作电压
D132	VIE	用于块擦除的 VDD	4.5	—	5.5	V	使用 ICSP™ 端口
D132A	VIW	外部定时擦写的 VDD	4.5	—	5.5	V	使用 ICSP 端口
D132B	VPEW	自定时擦写的 VDD	VMIN	—	5.5	V	VMIN = 最小工作电压
D133	TIE	ICSP 块擦除周期时间	—	4	—	ms	VDD > 4.5V
D133A	TIW	ICSP 擦写周期时间（外部定时）	2	—	—	ms	VDD > 4.5V
D133A	TIW	自定时写周期时间	—	2	—	ms	
D134	TRETD	保存时间	40	100	—	年	假设没有违反其他规范

† 除非另行声明，否则“典型值”栏中的数据均为 5.0V，25°C 条件下的值。这些参数仅供设计参考，未经测试。

表 26-2： 比较器规范

工作条件：除非另行声明，否则均为 3.0V < VDD < 5.5V， -40°C < TA < +85°C。							
参数编号	符号	特性	最小值	典型值	最大值	单位	备注
D300	VIOFF	输入失调电压	—	± 5.0	± 10	mV	
D301	VICM	输入共模电压 *	0	—	VDD – 1.5	V	
D302	CMRR	共模抑制比 *	55	—	—	dB	
300	TRESP	响应时间 （1） *	—	150	400	ns	PIC18**F**XXX
300A			—	150	600	ns	PIC18**LF**XXXX，VDD = 2.0V
301	TMC2OV	比较器模式变为输出有效 *	—	—	10	µs	

* 这些参数仅为特征值，未经测试。

注 1： 响应时间是在比较器的一个输入端的电压为 （VDD – 1.5） /2，而另一个输入端从 VSS 跳变到 VDD 时测得的。

表 26-3： 参考电压规范

工作条件：除非另行声明，否则均为 3.0V < VDD < 5.5V， -40°C< TA < +85°C。							
参数编号	符号	特性	最小值	典型值	最大值	单位	备注
D310	VRES	分辨率	VDD/24	—	VDD/32	LSb	
D311	VRAA	绝对精度	—	—	1/2	LSb	
D312	VRUR	单位电阻值 （R）	—	2k	—	Ω	
310	TSET	稳定时间 (1)	—	—	10	µs	

注 1： 稳定时间是在 CVRR = 1 并且 CVR3:CVR0 从 0000 转跳变到 1111 时测得的。

图 26-3： 高 / 低压检测特性

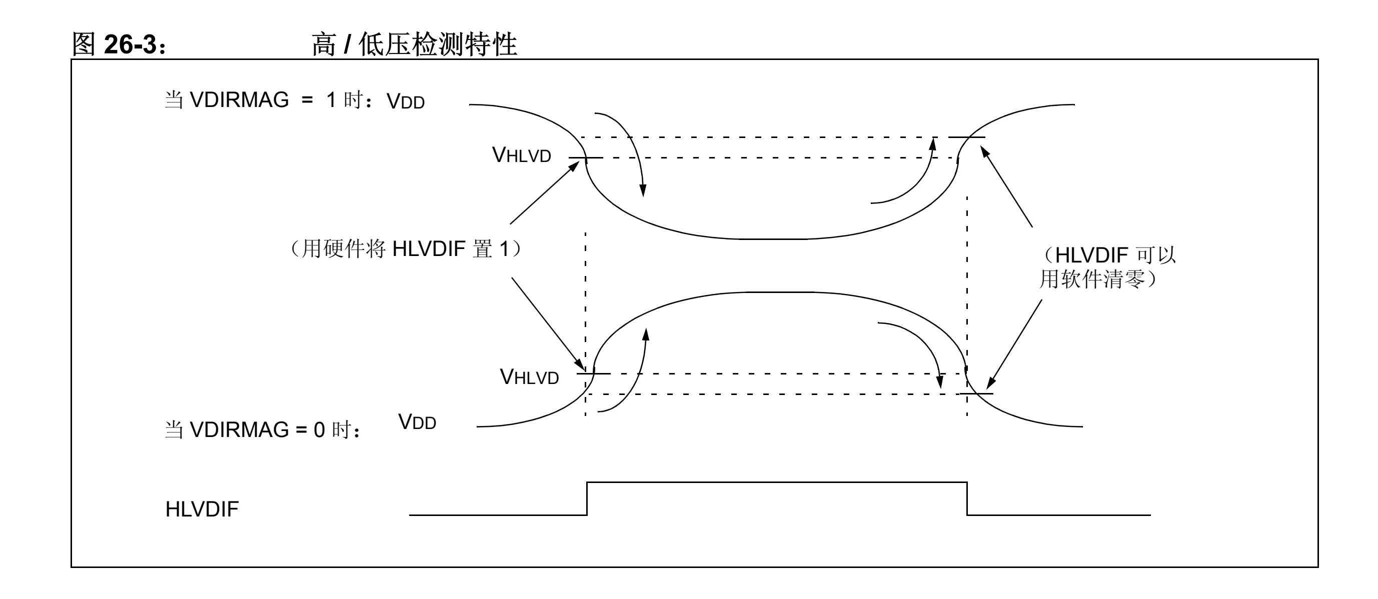


表 26-4： 低压检测规范

				标准工作条件 （除非另行声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级）				
参数编号	符号	特性		最小值	典型值 †	最大值	单位	条件
D420		VDD 由高转变为低时的 HLVD 电压	LVV = 0000	2.06	2.17	2.28	V	
			LVV = 0001	2.12	2.23	2.34	V	
			LVV = 0010	2.24	2.36	2.48	V	
			LVV = 0011	2.32	2.44	2.56	V	
			LVV = 0100	2.47	2.60	2.73	V	
			LVV = 0101	2.65	2.79	2.93	V	
			LVV = 0110	2.74	2.89	3.04	V	
			LVV = 0111	2.96	3.12	3.28	V	
			LVV = 1000	3.22	3.39	3.56	V	
			LVV = 1001	3.37	3.55	3.73	V	
			LVV = 1010	3.52	3.71	3.90	V	
			LVV = 1011	3.70	3.90	4.10	V	
			LVV = 1100	3.90	4.11	4.32	V	
			LVV = 1101	4.11	4.33	4.55	V	
			LVV = 1110	4.36	4.59	4.82	V	

† TAMB = 25°C 时对产品进行的测试。超过温度限制的规范由器件特性保证。

26.4 交流 （时序）规范

26.4.1 时序参数符号

可根据以下任一格式创建时序参数符号：

1. TppS2ppS		3. TCC:ST	（仅用于 I^{2}C 规范）
2. TppS		4. Ts	（仅用于 I^{2}C 规范）

T			
F	频率	T	时间

小写字母 （pp）及其含意：

pp			
cc	CCP1	osc	OSC1
ck	CLKO	rd	$\overline{RD}$
cs	$\overline{CS}$	rw	$\overline{RD}$ 或 $\overline{WR}$
di	SDI	sc	SCK
do	SDO	ss	$\overline{SS}$
dt	数据输入	t0	T0CKI
io	I/O 端口	t1	T13CKI
mc	$\overline{MCLR}$	wr	$\overline{WR}$

大写字母及其含意：

S			
F	下降	P	周期
H	高	R	上升
I	无效 （高阻态）	V	有效
L	低	Z	高阻态
仅用于 I^{2}C 模式			
AA	输出通道	High	高
BUF	总线空闲	Low	低

TCC:ST （仅用于 I^{2}C 规范）

CC			
HD	保持	SU	建立
施密特触发器			
DAT	保持数据输入	STO	停止条件
STA	启动条件		

26.4.2 时序条件

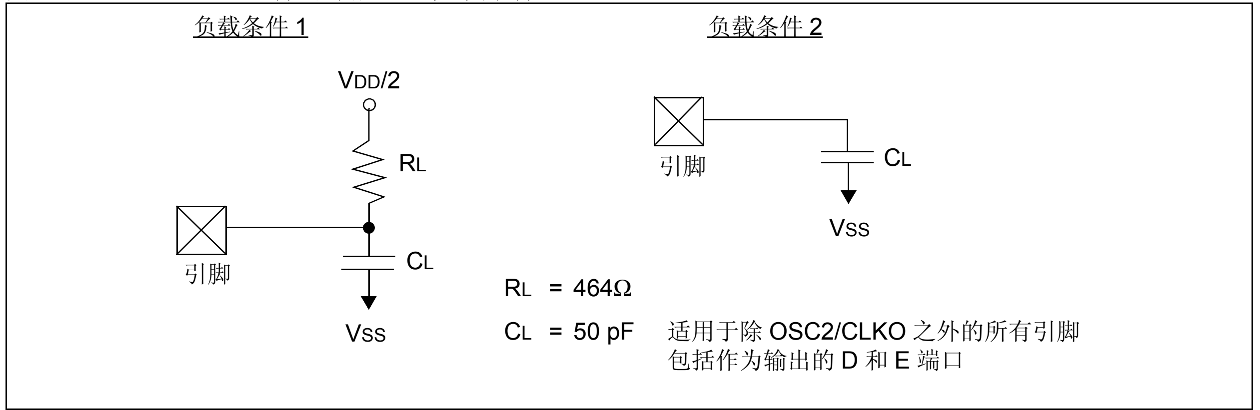
表 26-5 中指定的温度和电压适用于所有的时序规范（除非另行指明）。图 26-4 规定了时序规范的负载条件。

> **注：** 由于篇幅所限，本节中通称的“PIC18FXXXX”和“PIC18LFXXXX”特指（而且仅指代）PIC18F6390/6490/8390/8490 和 PIC18LF6390/6490/8390/8490 系列器件。

表 26-5： 温度和电压规范—交流

交流规范	**标准工作条件 （除非另行声明）** 工作温度 -40°C ≤ T_A ≤ +85°C （工业级） 直流规范第 **26.1** 节和第 **26.3** 节描述了工作电压 V_{DD} 的范围。 LF 器件仅在工业级温度下工作。

图 26-4： 器件时序规范的负载条件



26.4.3 时序图和规范

图 26-5： 外部时钟时序（除 PLL 之外的所有模式）

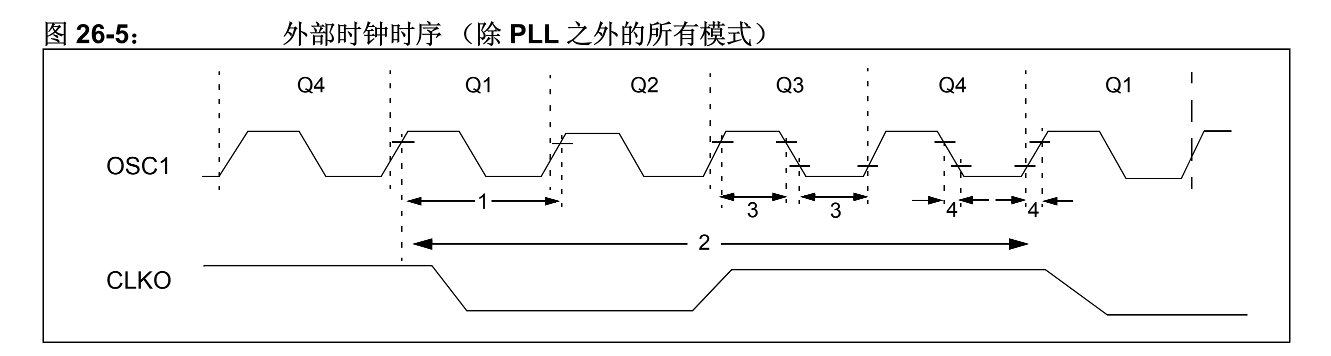


表 26-6： 外部时钟时序要求

参数编号	符号	特性	最小值	最大值	单位	条件
1A	FOSC	外部 CLKI 频率 (1)	DC	1	MHz	XT 和 RC 振荡器模式
			DC	20	MHz	HS 振荡器模式
			DC	31.25	kHz	LP 振荡器模式
		振荡器频率 (1)	DC	4	MHz	RC 振荡器模式
			0.1	4	MHz	XT 振荡器模式
			4	20	MHz	HS 振荡器模式
			5	200	kHz	LP 振荡器模式
1	TOSC	外部时钟 CLKI 周期 (1)	1000	—	ns	XT 和 RC 振荡器模式
			50	—	ns	HS 振荡器模式
			32	—	μs	LP 振荡器模式
		振荡器周期 (1)	250	—	ns	RC 振荡器模式
			250	1	μs	XT 振荡器模式
			100	250	ns	HS 振荡器模式
			50	250	ns	HS 振荡器模式
			5	—	μs	LP 振荡器模式
2	TCY	指令周期时间 (1)	100	—	ns	TCY = 4/FOSC
3	TOSL，TOSH	外部时钟输入（OSC1）的高电平或低电平时间	30	—	ns	XT 振荡器
			2.5	—	μs	LP 振荡器
			10	—	ns	HS 振荡器
4	TOSR，TOSF	外部时钟输入（OSC1）的上升或下降时间	—	20	ns	XT 振荡器
			—	50	ns	LP 振荡器
			—	7.5	ns	HS 振荡器

注 1： 对于除 PLL 外的所有配置，指令周期时间（TCY）等于输入振荡器时基周期的四倍。所有值均基于器件在标准工作条件下执行代码所对应的特定振荡器类型的特征数据。超过规定值可能导致振荡器运行不稳定和 / 或电流消耗超出预期。所有器件在测试 “最小值”时，都在 OSC1/CLKI 引脚连接了外部时钟。当使用了外部时钟输入时，所有器件的 “最大”周期时间限制为 “DC”（没有时钟）。

 初稿

表 26-7： PLL 时钟时序规范 （VDD = 4.2V 至 5.5V）

参数编号	符号	特性	最小值	典型值	最大值	单位	条件
F10	FOSC	振荡器频率范围	4	—	10	MHz	仅 HS 模式
F11	FSYS	片上 VCO 系统频率	16	—	40	MHz	仅 HS 模式
F12	t_{rc}	PLL 起振时间 （锁定时间）	—	—	2	ms	
F13	DCLK	CLKO 稳定性 （抗抖动）	-2	—	+2	%	

† “典型值”栏中的数据均在 5.0V、 25°C 下测得 （除非另行声明）。这些参数仅供设计参考，未经测试。

表 26-8： 交流规范： 内部 RC 精度
PIC18LF6390/6490/8390/8490 （工业级）
PIC18F6390/6490/8390/8490 （工业级）

PIC18LF6390/6490/8390/8490 （工业级）	标准工作条件 （除非另行声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级）					
PIC18F6390/6490/8390/8490 （工业级）	标准工作条件 （除非另行声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级）					
参数编号	**器件**	**最小值**	**典型值**	**最大值**	**单位**	**条件**
	在 频率为 8 MHz、4 MHz、2 MHz、1 MHz、500 kHz、250 kHz 和 125 kHz[(1)] 时的 INTOSC 精度					
	PIC18LF6390/6490/8390/8490	-2	+/-1	2	%	+25°C VDD = 2.7-3.3 V
		-5	—	5	%	-10°C至+85°C VDD = 2.7-3.3 V
		-10	+/-1	10	%	-40°C 至 +85°C VDD = 2.7-3.3 V
	PIC18F6390/6490/8390/8490	-2	+/-1	2	%	+25°C VDD = 4.5-5.5 V
		-5	—	5	%	-10°C至 +85°C VDD = 4.5-5.5 V
		-10	+/-1	10	%	-40°C 至 +85°C VDD = 4.5-5.5 V
	在频率为 31 kHz[(2)] 时的 INTRC 精度					
	PIC18LF6390/6490/8390/8490	26.562	—	35.938	kHz	-40°C 至 +85°C VDD = 2.7-3.3 V
	PIC18F6390/6490/8390/8490	26.562	—	35.938	kHz	-40°C 至 +85°C VDD = 4.5-5.5 V

图注： 阴影行是为了增强表的可读性。
注 1： 频率校准温度为 25°C。 OSCTUNE 寄存器可用于补偿温度漂移。
2： 校准后的 INTRC 频率。

图 26-6： CLKO 和 I/O 时序

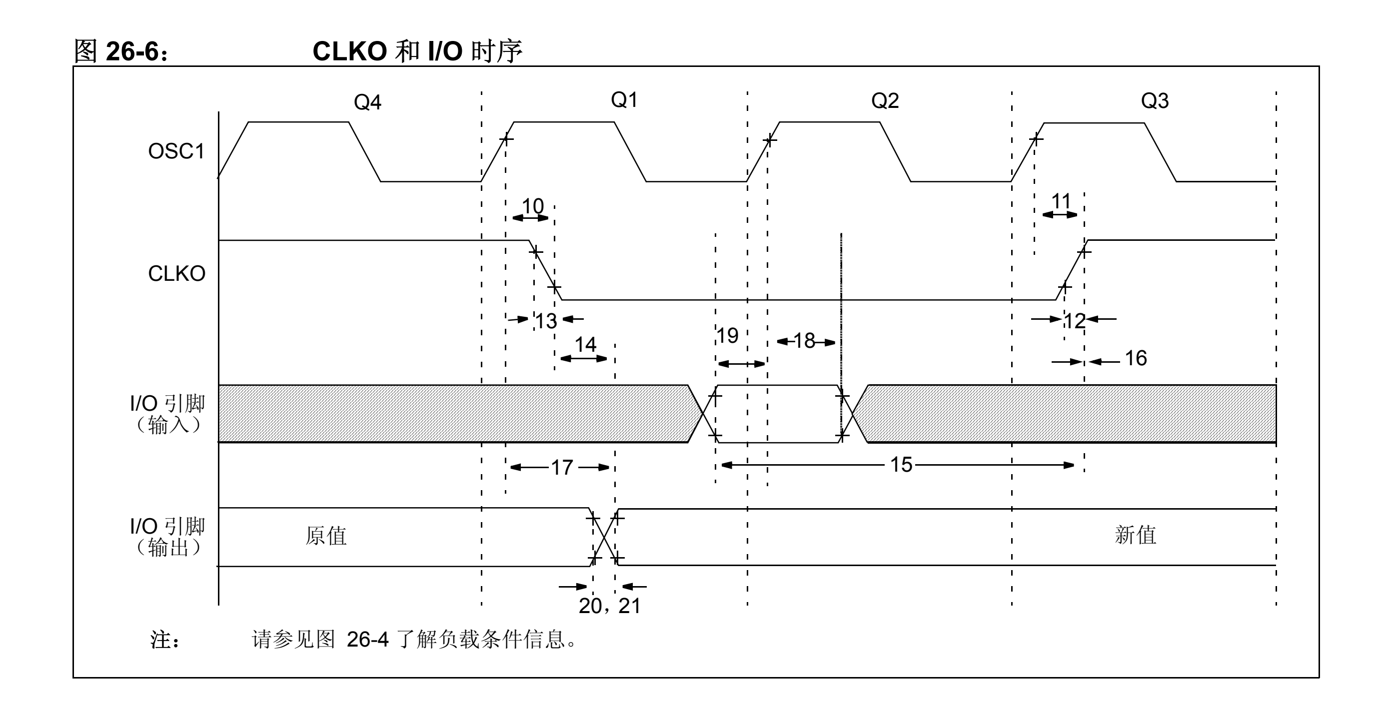


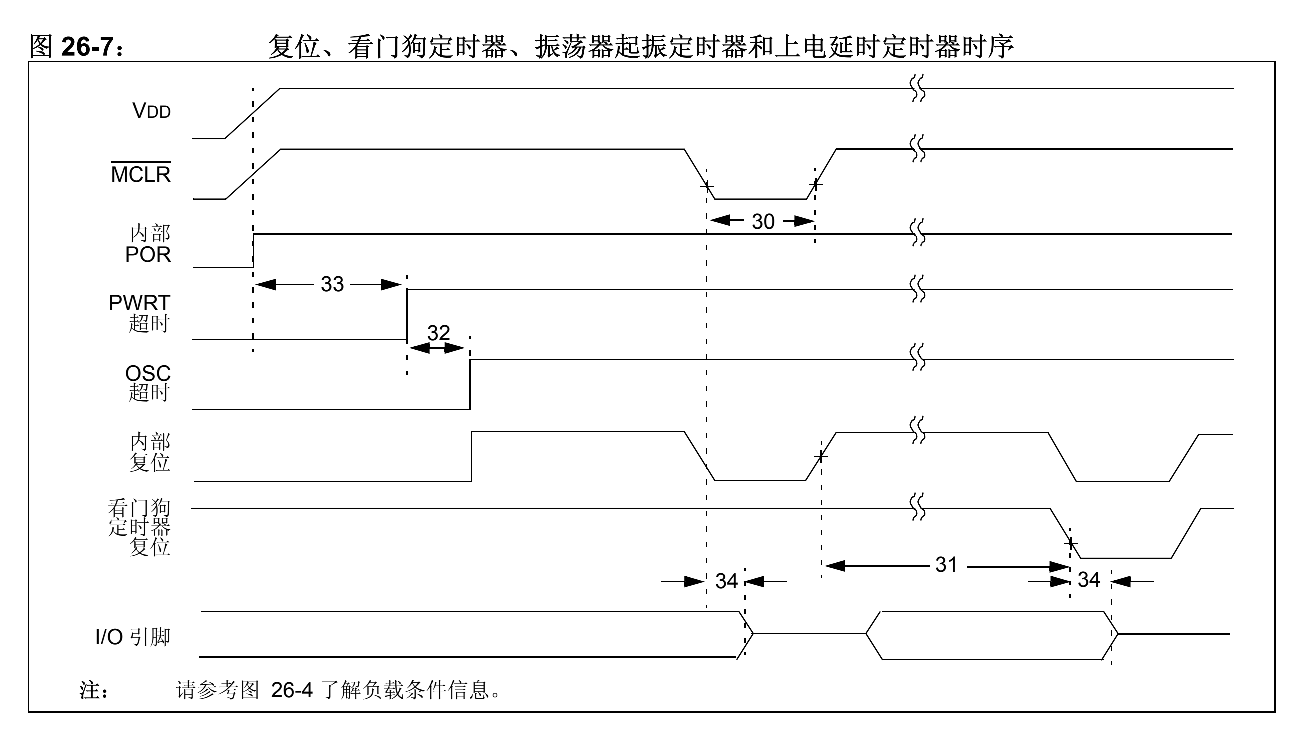
表 26-9： CLKO 和 I/O 时序要求

参数编号	符号	特性		最小值	典型值	最大值	单位	条件
10	TosH2ckL	OSC1 ↑ 到 CLKO ↓		—	75	200	ns	（注 1）
11	TosH2ckH	OSC1 ↑ 到 CLKO ↑		—	75	200	ns	（注 1）
12	TckR	CLKO 上升时间		—	35	100	ns	（注 1）
13	TckF	CLKO 下降时间		—	35	100	ns	（注 1）
14	TckL2ioV	CLKO ↓ 至端口输出有效		—	—	0.5 TCY + 20	ns	（注 1）
15	TioV2ckH	CLKO ↑ 之前端口输入有效		0.25 TCY +25	—	—	ns	（注 1）
16	TckH2ioI	CLKO ↑ 之后端口输入保持		0	—	—	ns	（注 1）
17	TosH2ioV	OSC1↑（Q1 周期）到端口输出有效		—	50	150	ns	
18	TosH2ioI	OSC1↑ （Q2 周期）到端口输入无效（I/O 输入保持时间）	PIC18**F**XXX	100	—	—	ns	
18A			PIC18**LF**XXX	200	—	—	ns	VDD = 2.0V
19	TioV2osH	端口输入有效到 OSC1↑（I/O 输入建立时间）		0	—	—	ns	
20	TioR	端口输出上升时间	PIC18**F**XXX	—	10	25	ns	
20A			PIC18**LF**XXX	—	—	60	ns	VDD = 2.0V
21	TioF	端口输出下降时间	PIC18**F**XXX	—	10	25	ns	
21A			PIC18**LF**XXX	—	—	60	ns	VDD = 2.0V
22†	TINP	INT 引脚高电平或低电平时间		TCY	—	—	ns	
23†	TRBP	RB7:RB4 改变 INT 高电平或低电平时间		TCY	—	—	ns	

† 这些参数是与任何内部时钟边沿无关的异步事件。

注 1： 测量是在 RC 模式下进行的，其中 CLKO 输出为 4 x TOSC。

图 26-7： 复位、看门狗定时器、振荡器起振定时器和上电延时定时器时序



注： 请参考图 26-4 了解负载条件信息。

图 26-8： 欠压复位时序

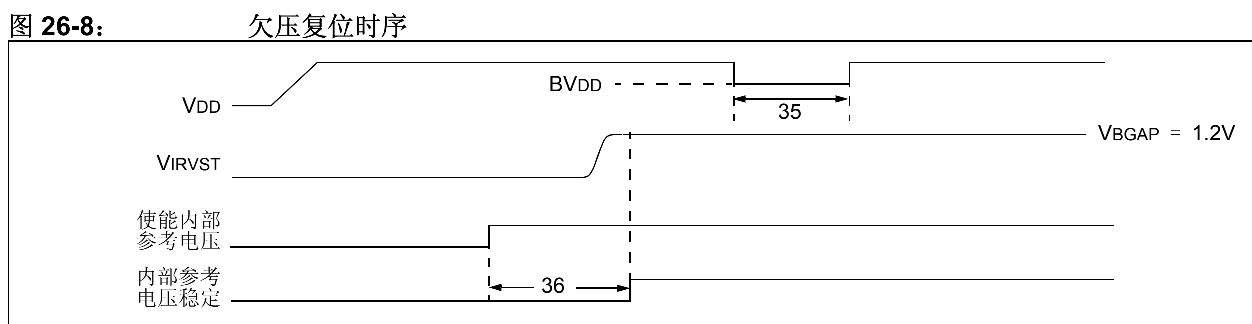


表 26-10： 复位、看门狗定时器、振荡器起振定时器、上电延时定时器和欠压复位要求

参数编号	符号	特性	最小值	典型值	最大值	单位	条件
30	TMCL	MCLR 脉冲宽度 （低电平）	2	—	—	µs	
31	TWDT	看门狗定时器超时溢出周期 （无后分频器）	3.4	4.0	4.6	ms	
32	TOST	振荡器起振定时器周期	1024 TOSC	—	1024 TOSC	—	TOSC = OSC1 周期
33	TPWRT	上电延时定时器周期	55.5	65.5	TBD	ms	
34	TIOZ	自 MCLR 低电平或看门狗定时器复位起 I/O 处于高阻态的时间	—	2	—	µs	
35	TBOR	欠压复位脉冲宽度	200	—	—	µs	VDD ≤ BVDD （见 D005）
36	TIVRST	内部参考电压稳定时间	—	20	50	µs	
37	TLVD	低压检测脉冲宽度	200	—	—	µs	VDD ≤ VLVD
38	TCSD	CPU 起振时间	—	10	—	µs	
39	TIOBST	INTRC 电路稳定时间	—	1	—	ms	

图注： TBD = 待定

图 26-9： TIMER0 和 TIMER1 外部时钟时序

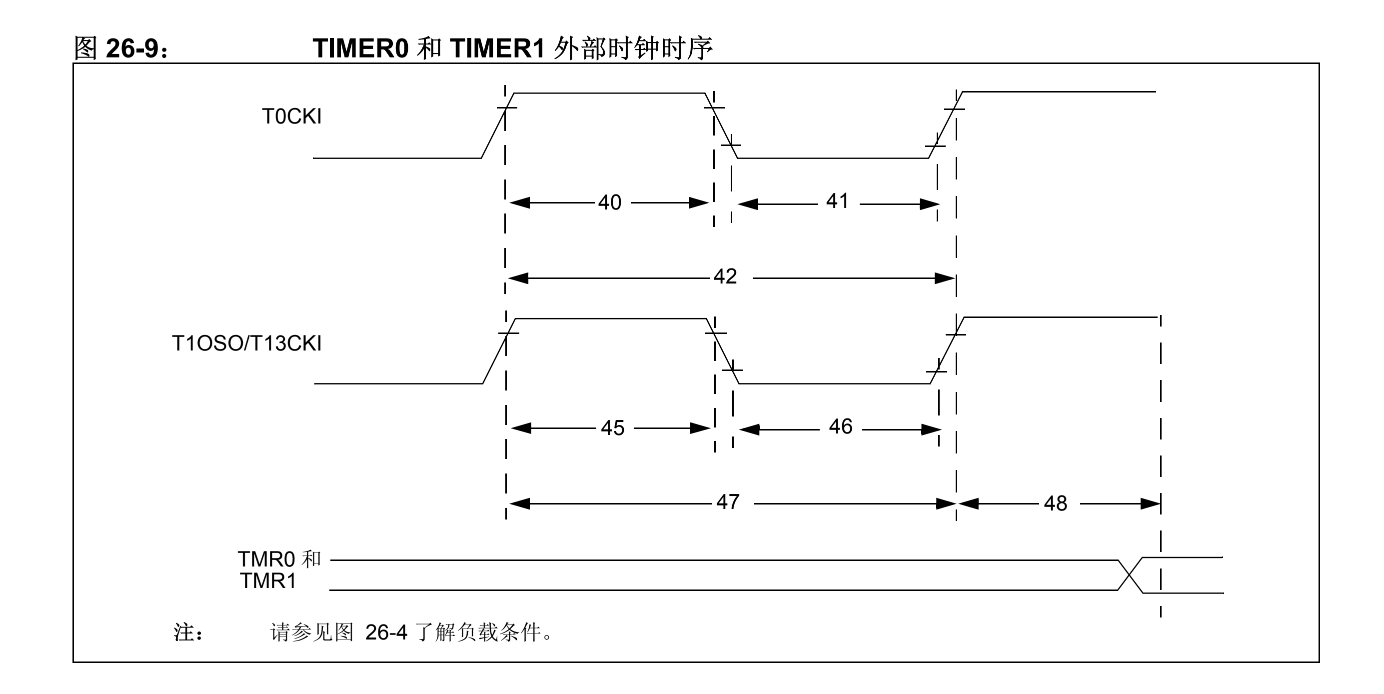
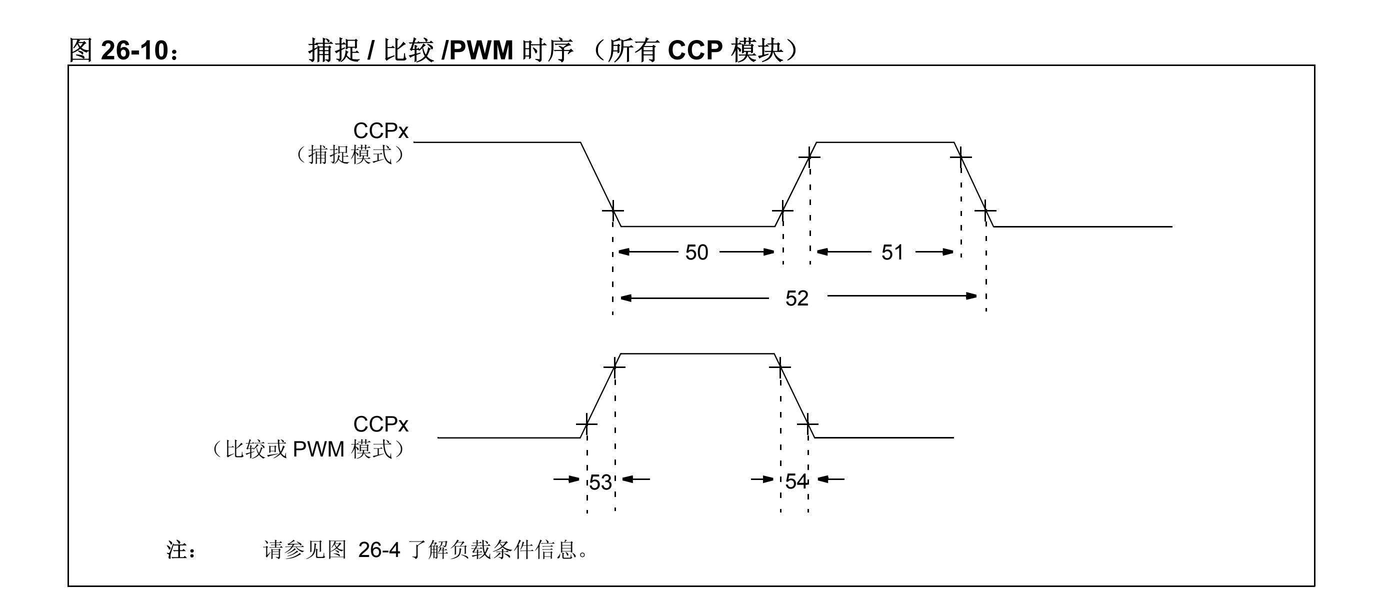


表 26-11： TIMER0 和 TIMER1 外部时钟要求

参数编号	符号	特性			最小值	最大值	单位	条件
40	TT0H	T0CKI 高电平脉冲宽度		无预分频器	0.5 TCY + 20	—	ns	
				有预分频器	10	—	ns	
41	TT0L	T0CKI 低电平脉冲宽度		无预分频器	0.5 TCY + 20	—	ns	
				有预分频器	10	—	ns	
42	TT0P	T0CKI 周期		无预分频器	TCY +10	—	ns	
				有预分频器	取较大值：20 ns 或 (TCY + 40)/N	—	ns	N = 预分频值（1、2、4、……256）
45	TT1H	T13CKI 高电平时间	同步，无预分频器		0.5 TCY + 20	—	ns	
			同步，有预分频器	PIC18**F**XXX	10	—	ns	
				PIC18**LF**XXX	25	—	ns	VDD = 2.0V
			异步	PIC18**F**XXX	30	—	ns	
				PIC18**LF**XXX	50	—	ns	VDD = 2.0V
46	TT1L	T13CKI 低电平时间	同步，无预分频器		0.5 TCY +5	—	ns	
			同步，有预分频器	PIC18**F**XXX	10	—	ns	
				PIC18**LF**XXX	25	—	ns	VDD = 2.0V
			异步	PIC18**F**XXX	30	—	ns	
				PIC18**LF**XXX	50	—	ns	VDD = 2.0V
47	TT1P	T13CKI 输入周期	同步		取较大值：20 ns 或 (TCY + 40)/N	—	ns	N = 预分频值 (1、 2、4 和 8)
			异步		60	—	ns	
	FT1	T13CKI 振荡器输入频率范围			DC	50	kHz	
48	TCKE2TMRI	从外部 T1CKI 时钟边沿到定时器加 1 的延时			2 TOSC	7 TOSC	—	

图 26-10： 捕捉 / 比较 /PWM 时序 （所有 CCP 模块）

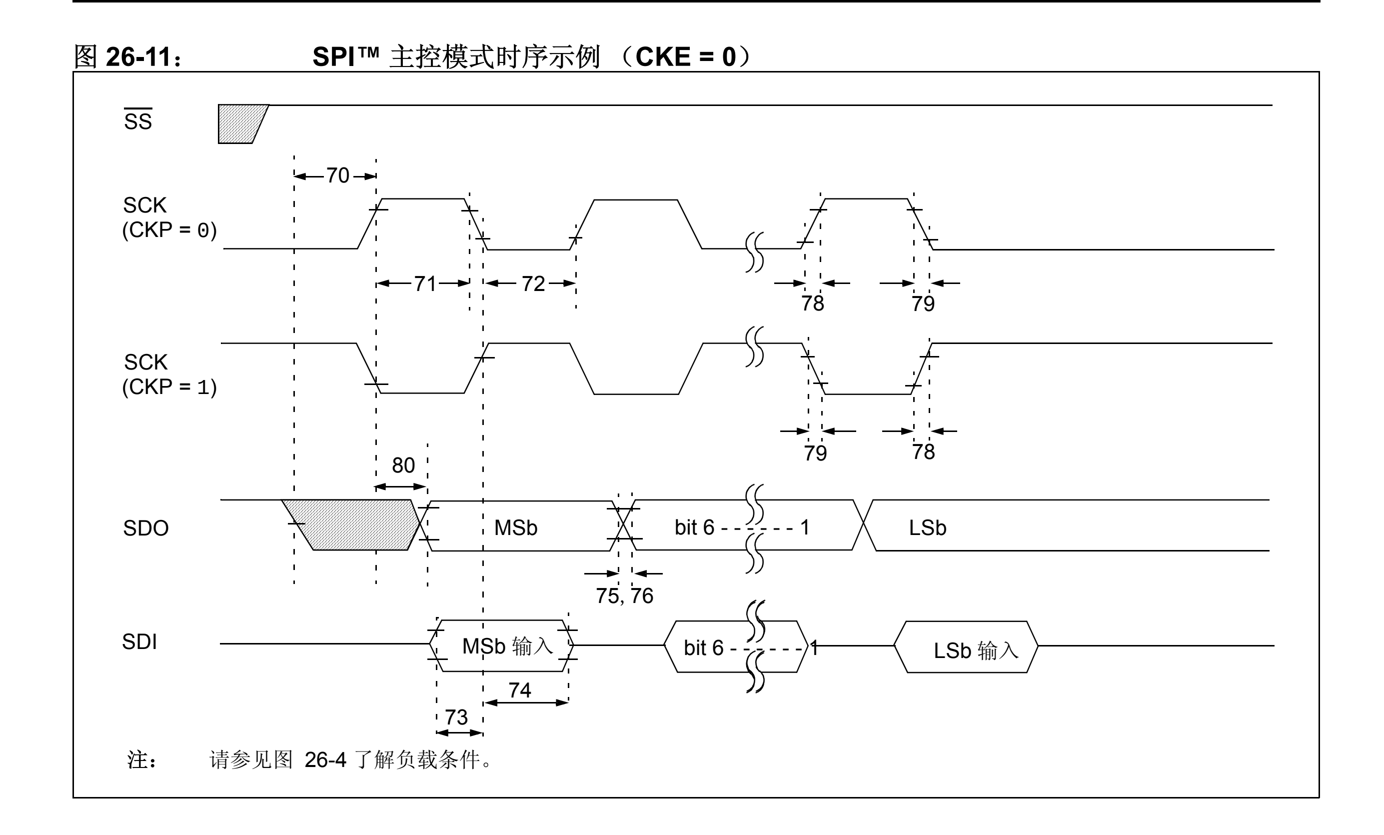


注： 请参见图 26-4 了解负载条件信息。

表 26-12： 捕捉 / 比较 /PWM 要求 （所有 CCP 模块）

参数编号	符号	特性			最小值	最大值	单位	条件
50	TCCL	CCPx 输入低电平时间	无预分频器		0.5 TCY + 20	—	ns	
			有预分频器	PIC18**F**XXX	10	—	ns	
				PIC18**LF**XXX	20	—	ns	VDD = 2.0V
51	TCCH	CCPx 输入高电平时间	无预分频器		0.5 TCY + 20	—	ns	
			有预分频器	PIC18**F**XXX	10	—	ns	
				PIC18**LF**XXX	20	—	ns	VDD = 2.0V
52	TCCP	CCPx 输入周期			$\frac{3 T_{CY} + 40}{N}$	—	ns	N = 预分频值（1、 4 或 16）
53	TCCR	CCPx 输出下降时间		PIC18**F**XXX	—	25	ns	
				PIC18**LF**XXX	—	45	ns	VDD = 2.0V
54	TCCF	CCPx 输出下降时间		PIC18**F**XXX	—	25	ns	
				PIC18**LF**XXX	—	45	ns	VDD = 2.0V

图 26-11： SPI™ 主控模式时序示例 （CKE = 0）



注： 请参见图 26-4 了解负载条件。

表 26-13： SPI™ 模式要求示例 （主控模式， CKE = 0）

参数编号	符号	特性		最小值	最大值	单位	条件
70	TssL2scH，TssL2scL	SS↓ 到 SCK↓ 或 SCK↑ 输入		TCY	—	ns	
71	TscH	SCK 输入高电平时间（从动模式）	连续	1.25 TCY + 30	—	ns	
71A			单字节	40	—	ns	（注 1）
72	TscL	SCK 输入低电平时间（从动模式）	连续	1.25 TCY + 30	—	ns	
72A			单字节	40	—	ns	（注 1）
73	TdiV2scH，TdiV2scL	SDI 数据输入到 SCK 边沿的建立时间		100	—	ns	
73A	TB2B	字节 1 的最后一个时钟边沿至字节 2 的第一个时钟边沿		1.5 TCY +40	—	ns	（注 2）
74	TscH2diL，TscL2diL	SDI 数据输入到 SCK 边沿的保持时间		100	—	ns	
75	TdoR	SDO 数据输出上升时间	PIC18**F**XXX	—	25	ns	
			PIC18**LF**XXX	—	45	ns	VDD = 2.0V
76	TdoF	SDO 数据输出下降时间		—	25	ns	
78	TscR	SCK 输出上升时间（主控模式）	PIC18**F**XXX	—	25	ns	
			PIC18**LF**XXX	—	45	ns	VDD = 2.0V
79	TscF	SCK 输出下降时间 （主控模式）		—	25	ns	
80	TscH2doV，TscL2doV	SCK 边沿后 SDO 数据输出有效	PIC18**F**XXX	—	50	ns	
			PIC18**LF**XXX	—	100	ns	VDD = 2.0V

注 **1**： 要求使用参数 #73A。
2： 仅当使用参数 #71A 和 #72A 时。

图 26-12：　　SPI™ 主控模式时序示例 （CKE = 1）

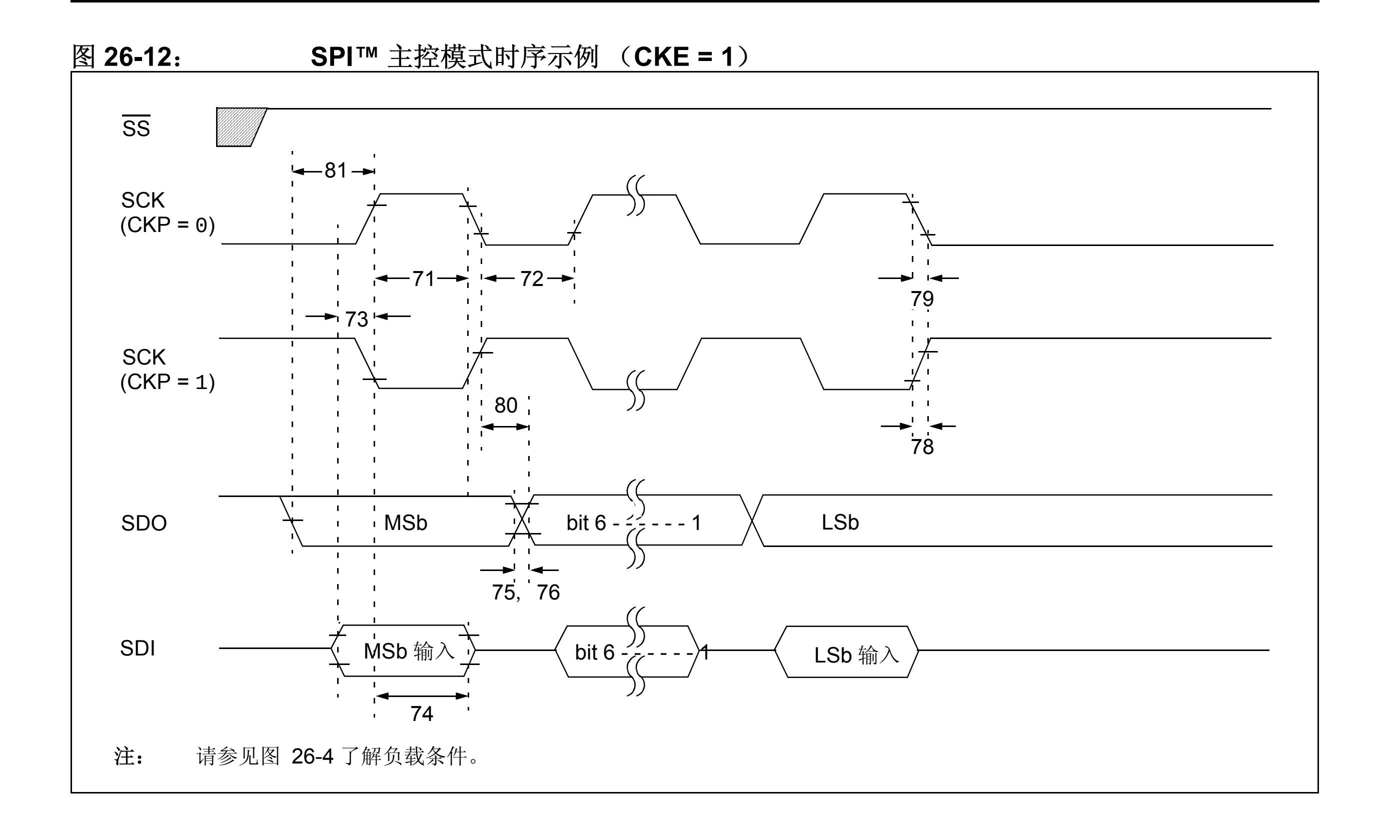


表 26-14：　　SPI™ 模式要求示例 （主控模式， CKE = 1）

参数编号	符号	特性		最小值	最大值	单位	条件
71	TscH	SCK 输入高电平时间（从动模式）	连续	1.25 TCY + 30	—	ns	
71A			单字节	40	—	ns	（注 **1**）
72	TscL	SCK 输入低电平时间（从动模式）	连续	1.25 TCY + 30	—	ns	
72A			单字节	40	—	ns	（注 **1**）
73	TdiV2scH， TdiV2scL	SDI 数据输入到 SCK 边沿的稳定时间		100	—	ns	
73A	TB2B	字节 1 的最后一个时钟边沿至字节 2 的第一个时钟边沿		1.5 TCY +40	—	ns	（注 **2**）
74	TscH2diL， TscL2diL	SDI 数据输入到 SCK 边沿的保持时间		100	—	ns	
75	TdoR	SDO 数据输出上升时间	PIC18**F**XXX	—	25	ns	
			PIC18**LF**XXX	—	45	ns	VDD = 2.0V
76	TdoF	SDO 数据输出下降时间		—	25	ns	
78	TscR	SCK 输出上升时间（主控模式）	PIC18**F**XXX	—	25	ns	
			PIC18**LF**XXX	—	45	ns	VDD = 2.0V
79	TscF	SCK 输出下降时间 （主控模式）		—	25	ns	
80	TscH2doV， TscL2doV	SCK 边沿后 SDO 数据输出有效	PIC18**F**XXX	—	50	ns	
			PIC18**LF**XXX	—	100	ns	VDD = 2.0V
81	TdoV2scH， TdoV2scL	SDO 数据输出到 SCK 边沿的建立时间		TCY	—	ns	

注　**1**： 要求使用参数 #73A。
　　2： 仅当使用参数 #71A 和 #72A 时。

图 26-13：SPI™ 从动模式时序示例 （CKE = 0）

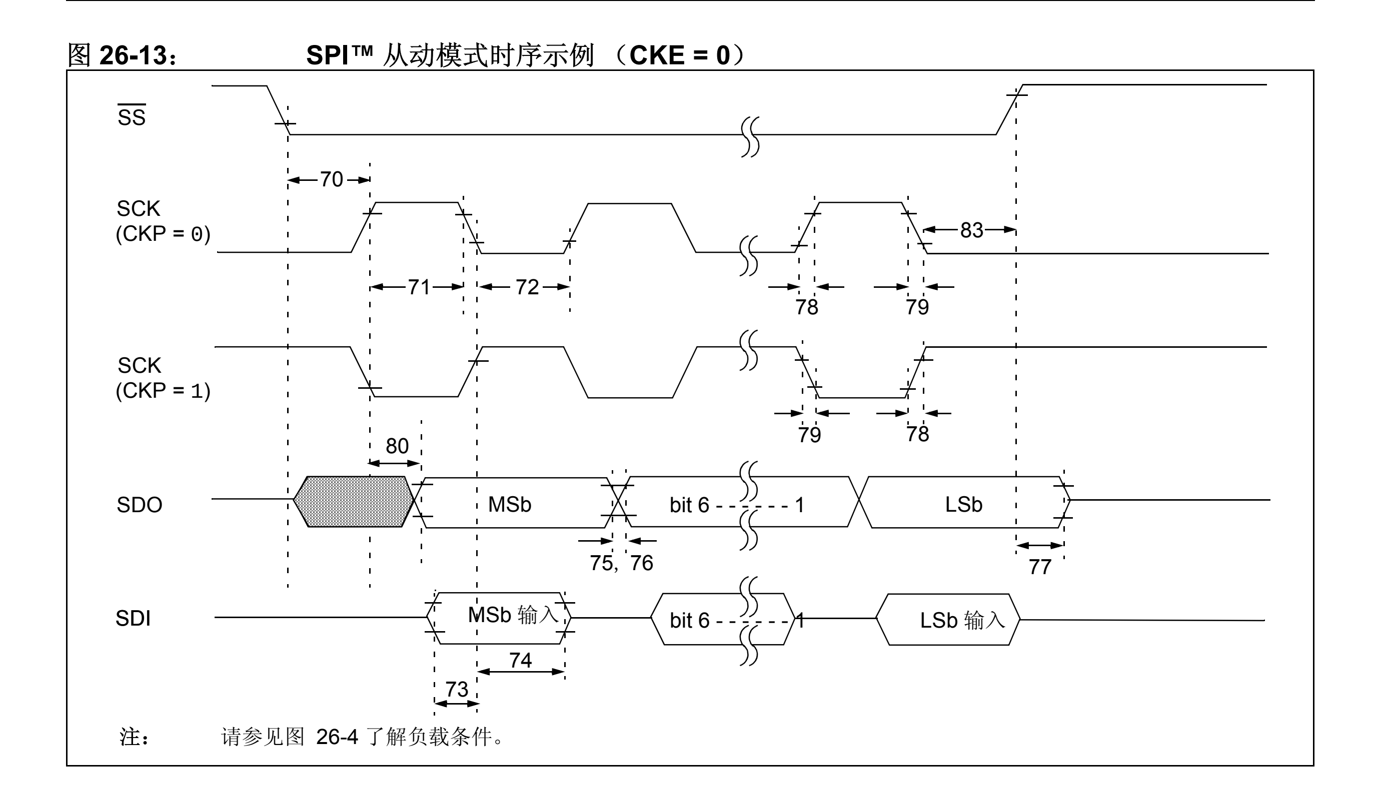
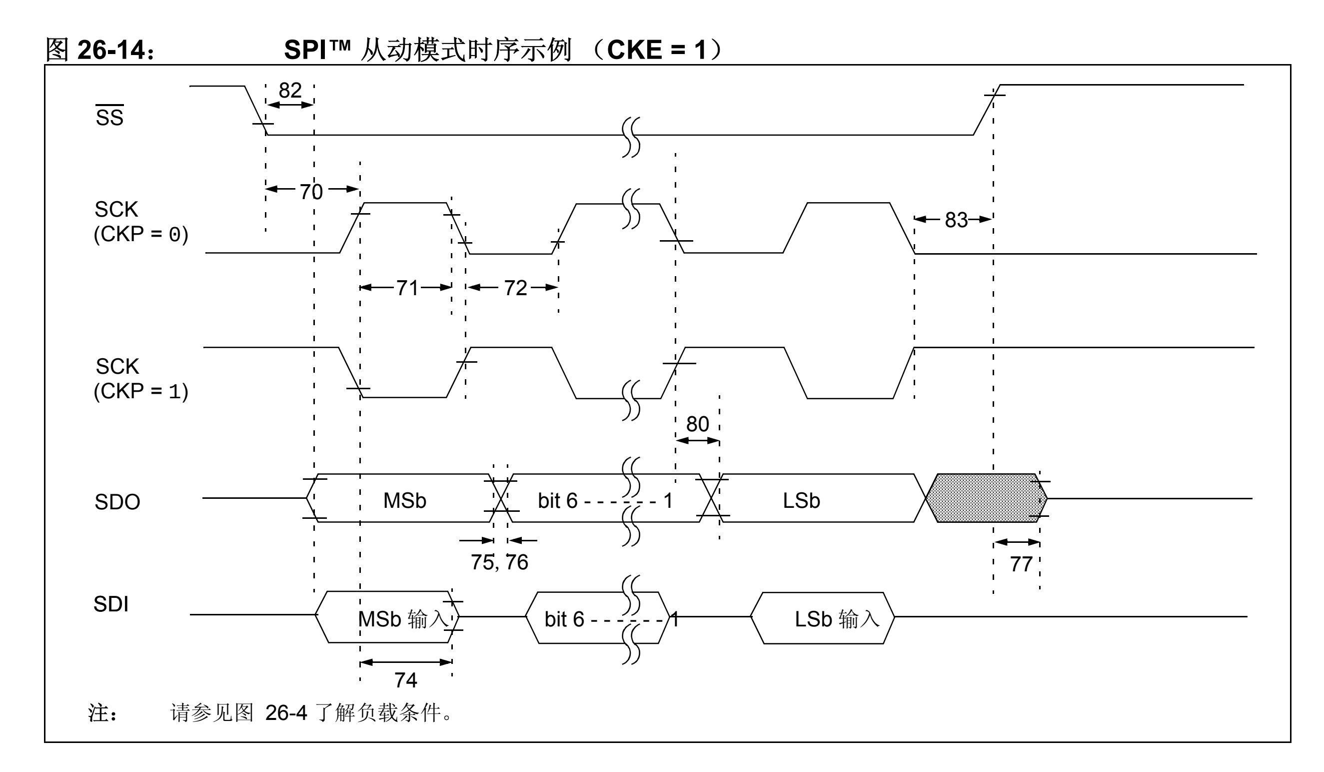


表 26-15：SPI™ 模式要求示例 （从动模式， CKE = 0）

参数编号	符号	特性		最小值	最大值	单位	条件
70	TssL2scH，TssL2scL	$\overline{SS}$↓ 到 SCK↓ 或 SCK↑ 输入		TCY	—	ns	
71	TscH	SCK 输入高电平时间（从动模式）	连续	1.25 TCY + 30	—	ns	
71A			单字节	40	—	ns	（注 1）
72	TscL	SCK 输入低电平时间（从动模式）	连续	1.25 TCY + 30	—	ns	
72A			单字节	40	—	ns	（注 1）
73	TdiV2scH，TdiV2scL	SDI 数据输入到 SCK 边沿的建立时间		100	—	ns	
73A	TB2B	字节 1 的最后一个时钟边沿至字节 2 的第一个时钟边沿		1.5 TCY +40	—	ns	（注 2）
74	TscH2diL，TscL2diL	SDI 数据输入到 SCK 边沿的保持时间		100	—	ns	
75	TdoR	SDO 数据输出上升时间	PIC18**F**XXX	—	25	ns	
			PIC18**LF**XXX	—	45	ns	VDD = 2.0V
76	TdoF	SDO 数据输出下降时间		—	25	ns	
77	TssH2doZ	$\overline{SS}$↑ 到 SDO 输出高阻抗		10	50	ns	
78	TscR	SCK 输出上升时间 （主控模式）	PIC18**F**XXX	—	25	ns	
			PIC18**LF**XXX	—	45	ns	VDD = 2.0V
79	TscF	SCK 输出下降时间 （主控模式）		—	25	ns	
80	TscH2doV，TscL2doV	SCK 边沿后 SDO 数据输出有效	PIC18**F**XXX	—	50	ns	
			PIC18**LF**XXX	—	100	ns	VDD = 2.0V
83	TscH2ssH，TscL2ssH	SCK 边沿到出现 $\overline{SS}$ ↑		1.5 TCY +40	—	ns	

注 **1**：要求使用参数 #73A。
2：仅当使用参数 #71A 和 #72A 时。

图 26-14： SPI™ 从动模式时序示例 （CKE = 1）



注： 请参见图 26-4 了解负载条件。

表 26-16： SPI™ 从动模式要求示例 （CKE = 1）

参数编号	符号	特性		最小值	最大值	单位	条件
70	TssL2scH，TssL2scL	SS↓ 到 SCK↓ 或 SCK↑ 输入		TCY	—	ns	
71	TscH	SCK 输入高电平时间（从动模式）	连续	1.25 TCY + 30	—	ns	
71A			单字节	40	—	ns	（注 1）
72	TscL	SCK 输入低电平时间（从动模式）	连续	1.25 TCY + 30	—	ns	
72A			单字节	40	—	ns	（注 1）
73A	TB2B	字节 1 的最后一个时钟边沿至字节 2 的第一个时钟边沿		1.5 TCY +40	—	ns	（注 2）
74	TscH2diL，TscL2diL	SDI 数据输入到 SCK 边沿的保持时间		100	—	ns	
75	TdoR	SDO 数据输出上升时间	PIC18**F**XXX	—	25	ns	
			PIC18**LF**XXX	—	45	ns	VDD = 2.0V
76	TdoF	SDO 数据输出下降时间		—	25	ns	
77	TssH2doZ	SS↑ 到 SDO 输出高阻抗		10	50	ns	
78	TscR	SCK 输出上升时间（主控模式）	PIC18**F**XXX	—	25	ns	
			PIC18**LF**XXX	—	45	ns	VDD = 2.0V
79	TscF	SCK 输出下降时间 （主控模式）		—	25	ns	
80	TscH2doV，TscL2doV	SCK 边沿后 SDO 数据输出有效	PIC18**F**XXX	—	50	ns	
			PIC18**LF**XXX	—	100	ns	VDD = 2.0V
82	TssL2doV	在 SS ↓边沿之后 SDO 数据输出有效	PIC18**F**XXX	—	50	ns	
			PIC18**LF**XXX	—	100	ns	VDD = 2.0V
83	TscH2ssH，TscL2ssH	在 SCK 边缘之后出现 SS ↑		1.5 TCY +40	—	ns	

注 **1**： 要求使用参数 #73A。
2： 仅当使用参数 #71A 和 #72A 时。

图 26-15： I^2C™ 总线启动 / 停止位时序

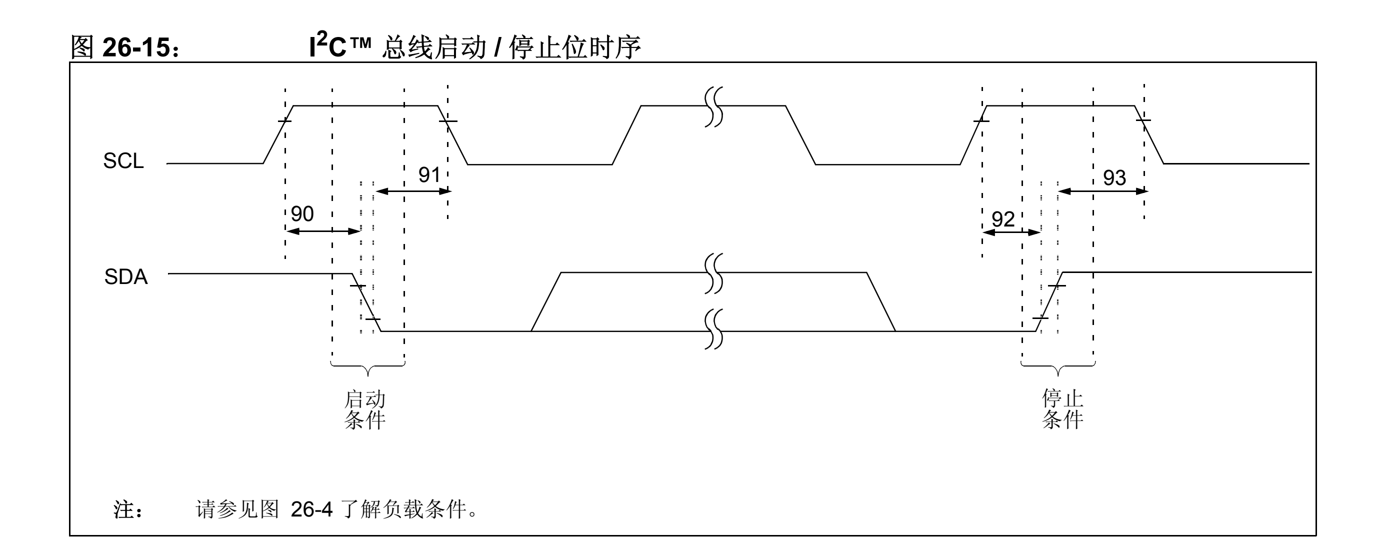


表 26-17： I^2C™ 总线启动 / 停止位要求 （从动模式）

参数编号	符号	特性		最小值	最大值	单位	条件
90	TSU:STA	启动条件	100 kHz 模式	4700	—	ns	仅与重复起始条件相关
		建立时间	400 kHz 模式	600	—		
91	THD:STA	启动条件	100 kHz 模式	4000	—	ns	这个周期后产生第一个时钟脉冲
		保持时间	400 kHz 模式	600	—		
92	TSU:STO	停止条件	100 kHz 模式	4700	—	ns	
		建立时间	400 kHz 模式	600	—		
93	THD:STO	停止条件	100 kHz 模式	4000	—	ns	
		保持时间	400 kHz 模式	600	—		

图 26-16： I^2C™ 总线数据时序

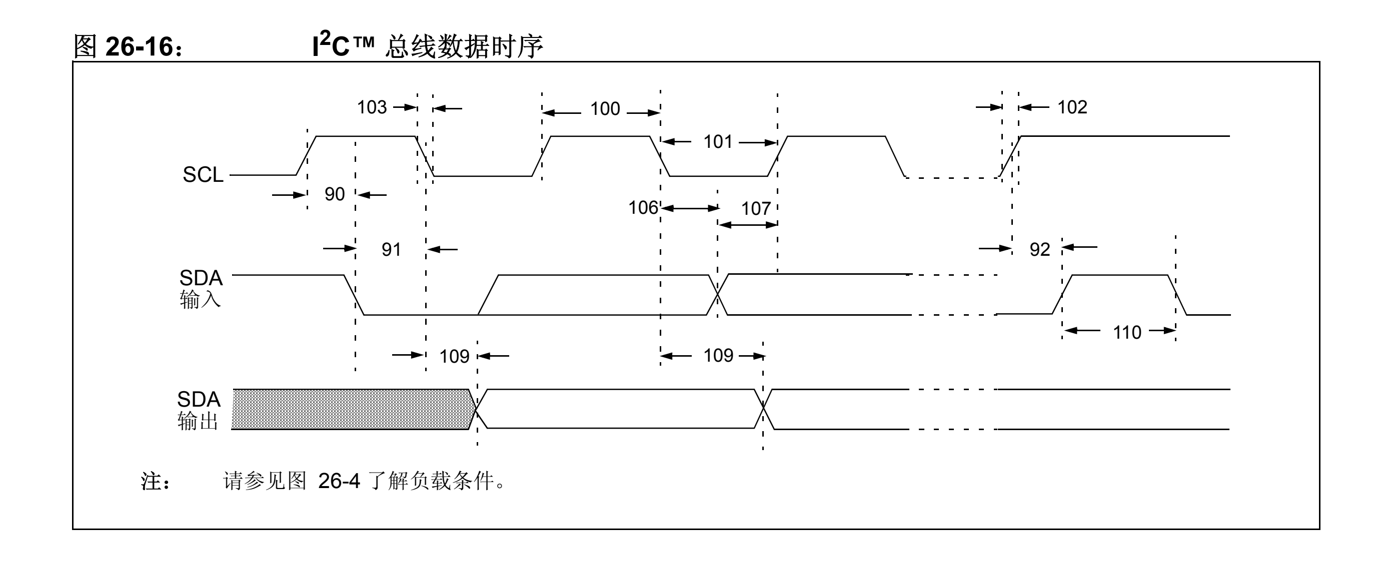


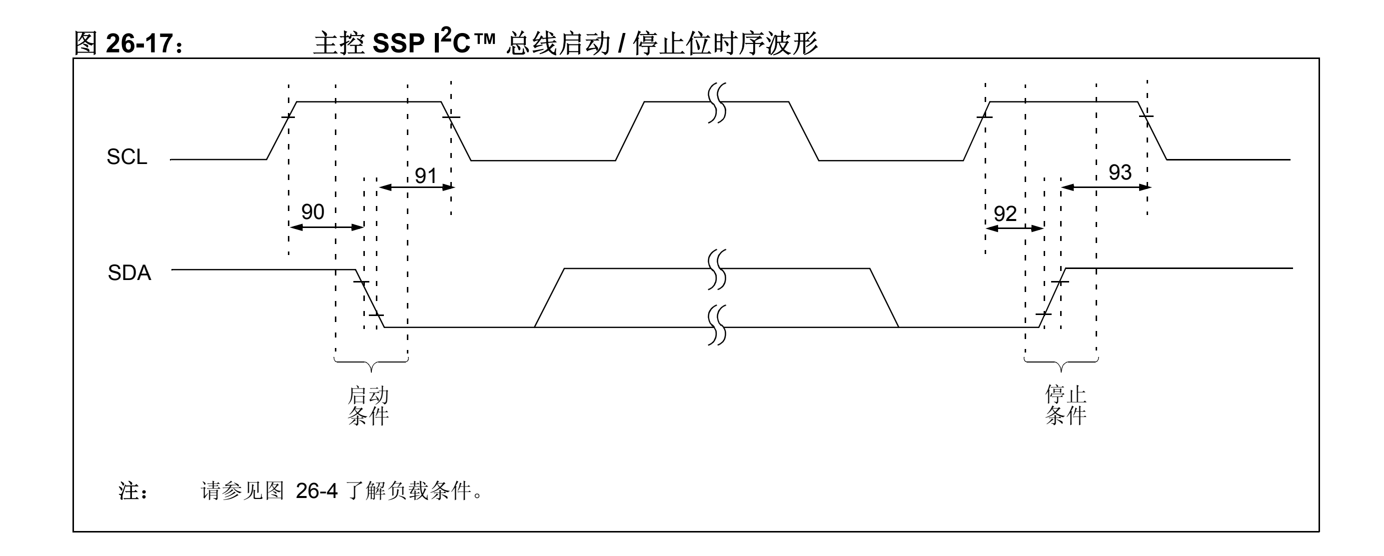
表 **26-18**： **I²C™** 总线数据要求 （从动模式）

参数编号	符号	特性		最小值	最大值	单位	条件
100	THIGH	时钟高电平时间	100 kHz 模式	4.0	—	μs	PIC18FXXX 工作频率不得低于 1.5 MHz
			400 kHz 模式	0.6	—	μs	PIC18FXXX 工作频率不得低于 10 MHz
			SSP 模块	1.5 TCY	—		
101	TLOW	时钟低电平时间	100 kHz 模式	4.7	—	μs	PIC18FXXX 工作频率不得低于 1.5 MHz
			400 kHz 模式	1.3	—	μs	PIC18FXXX 工作频率不得低于 10 MHz
			SSP 模块	1.5 TCY	—		
102	TR	SDA 和 SCL 上升时间	100 kHz 模式	—	1000	ns	
			400 kHz 模式	20 + 0.1 CB	300	ns	CB 值的范围在 10 至 400 pF 之间
103	TF	SDA 和 SCL 下降时间	100 kHz 模式	—	300	ns	
			400 kHz 模式	20 + 0.1 CB	300	ns	CB 值的范围在 10 至 400 pF 之间
90	TSU:STA	启动条件建立时间	100 kHz 模式	4.7	—	ms	仅与重复启动条件相关
			400 kHz 模式	0.6	—	ms	
91	THD:STA	启动条件保持时间	100 kHz 模式	4.0	—	ms	这个周期后产生第一个时钟脉冲
			400 kHz 模式	0.6	—	ms	
106	THD:DAT	数据输入保持时间	100 kHz 模式	0	—	ns	
			400 kHz 模式	0	0.9	ms	
107	TSU:DAT	数据输入建立时间	100 kHz 模式	250	—	ns	（注 **2**）
			400 kHz 模式	100	—	ns	
92	TSU:STO	停止条件建立时间	100 kHz 模式	4.7	—	ms	
			400 kHz 模式	0.6	—	ms	
109	TAA	时钟输出有效时间	100 kHz 模式	—	3500	ns	（注 **1**）
			400 kHz 模式	—	—	ns	
110	TBUF	总线空闲时间	100 kHz 模式	4.7	—	μs	在启动一个新的传输前总线保持空闲的时间
			400 kHz 模式	1.3	—	μs	
D102	CB	总线容性负载		—	400	pF	

注 **1**： 为避免产生意外的启动或停止条件，作为发送器的器件必须提供这个内部最小延时以覆盖 SCL 下降沿的未定义区域 （最小值 300ns）。

2： 快速模式的 I²C 总线器件也可在标准模式的 I²C™ 总线系统中使用，但必须满足 TSU:DAT ≥ 250 ns 的要求。如果快速模式器件没有延长 SCL 信号的低电平周期，则必然满足此条件。如果该器件延长了 SCL 信号的低电平周期，其下一个数据位必须输出到 SDA 线。SCL 线被释放前，根据标准模式 I²C 总线规范，TR max. + TSU:DAT = 1000 + 250 = 1250 ns。

图 26-17： 主控 SSP I^2C™ 总线启动 / 停止位时序波形



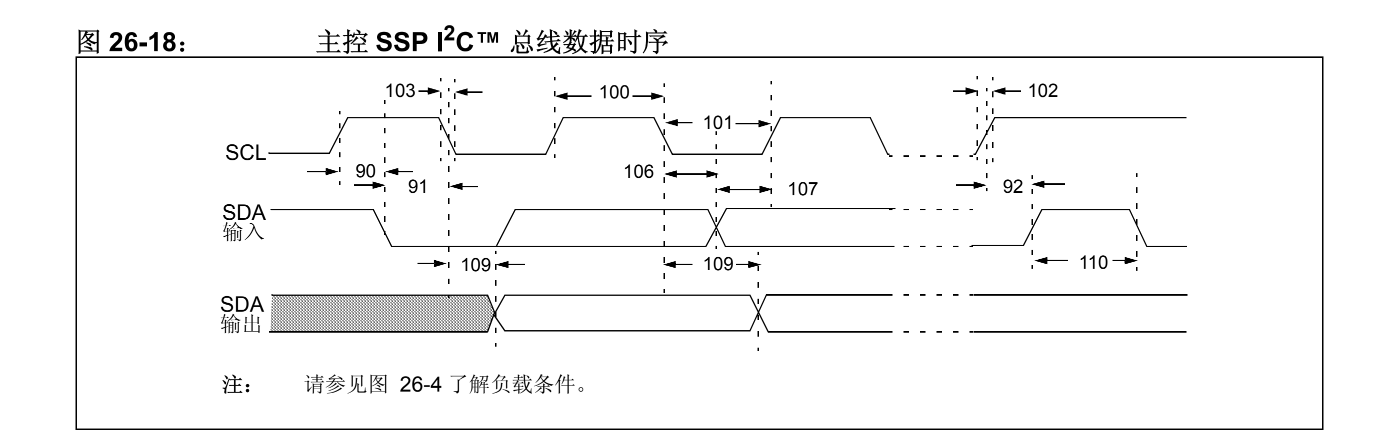
注： 请参见图 26-4 了解负载条件。

表 26-19： 主控 SSP I^2C™ 总线启动 / 停止位要求

参数编号	符号	特性		最小值	最大值	单位	条件
90	TSU:STA	启动条件建立时间	100 kHz 模式	2(TOSC)(BRG + 1)	—	ns	仅与重复起始条件相关
			400 kHz 模式	2(TOSC)(BRG + 1)	—		
			1 MHz 模式 (1)	2(TOSC)(BRG + 1)	—		
91	THD:STA	启动条件保持时间	100 kHz 模式	2(TOSC)(BRG + 1)	—	ns	这个周期后产生第一个时钟脉冲
			400 kHz 模式	2(TOSC)(BRG + 1)	—		
			1 MHz 模式 (1)	2(TOSC)(BRG + 1)	—		
92	TSU:STO	停止条件建立时间	100 kHz 模式	2(TOSC)(BRG + 1)	—	ns	
			400 kHz 模式	2(TOSC)(BRG + 1)	—		
			1 MHz 模式 (1)	2(TOSC)(BRG + 1)	—		
93	THD:STO	停止条件保持时间	100 kHz 模式	2(TOSC)(BRG + 1)	—	ns	
			400 kHz 模式	2(TOSC)(BRG + 1)	—		
			1 MHz 模式 (1)	2(TOSC)(BRG + 1)	—		

注 1： 对于所有 I^2C™ 引脚，最小引脚电容为 10 pF。

图 26-18： 主控 SSP I^2C™ 总线数据时序



注： 请参见图 26-4 了解负载条件。

初稿

表 **26-20**： 主控 **SSP I²C™** 总线数据要求

参数编号	符号	特性		最小值	最大值	单位	条件
100	THIGH	时钟高电平时间	100 kHz 模式	2(TOSC)(BRG + 1)	—	ms	
			400 kHz 模式	2(TOSC)(BRG + 1)	—	ms	
			1 MHz 模式 (1)	2(TOSC)(BRG + 1)	—	ms	
101	TLOW	时钟低电平时间	100 kHz 模式	2(TOSC)(BRG + 1)	—	ms	
			400 kHz 模式	2(TOSC)(BRG + 1)	—	ms	
			1 MHz 模式 (1)	2(TOSC)(BRG + 1)	—	ms	
102	TR	SDA 和 SCL 上升时间	100 kHz 模式	—	1000	ns	CB 值指在 10 至 400 pF 之间
			400 kHz 模式	20 + 0.1 CB	300	ns	
			1 MHz 模式 (1)	—	300	ns	
103	TF	SDA 和 SCL 下降时间	100 kHz 模式	—	300	ns	CB 值在 10 至 400 pF 之间
			400 kHz 模式	20 + 0.1 CB	300	ns	
			1 MHz 模式 (1)	—	100	ns	
90	TSU:STA	启动条件建立时间	100 kHz 模式	2(TOSC)(BRG + 1)	—	ms	仅与重复启动条件相关
			400 kHz 模式	2(TOSC)(BRG + 1)	—	ms	
			1 MHz 模式 (1)	2(TOSC)(BRG + 1)	—	ms	
91	THD:STA	启动条件保持时间	100 kHz 模式	2(TOSC)(BRG + 1)	—	ms	这个周期后产生第一个时钟脉冲
			400 kHz 模式	2(TOSC)(BRG + 1)	—	ms	
			1 MHz 模式 (1)	2(TOSC)(BRG + 1)	—	ms	
106	THD:DAT	数据输入保持时间	100 kHz 模式	0	—	ns	
			400 kHz 模式	0	0.9	ms	
			1 MHz 模式 (1)	TBD	—	ns	
107	TSU:DAT	数据输入建立时间	100 kHz 模式	250	—	ns	（注 **2**）
			400 kHz 模式	100	—	ns	
			1 MHz 模式 (1)	TBD	—	ns	
92	TSU:STO	停止条件建立时间	100 kHz 模式	2(TOSC)(BRG + 1)	—	ms	
			400 kHz 模式	2(TOSC)(BRG + 1)	—	ms	
			1 MHz 模式 (1)	2(TOSC)(BRG + 1)	—	ms	
109	TAA	时钟输出有效时间	100 kHz 模式	—	3500	ns	
			400 kHz 模式	—	1000	ns	
			1 MHz 模式 (1)	—	—	ns	
110	TBUF	总线空闲时间	100 kHz 模式	4.7	—	ms	在开始一个新的传输前总线保持空闲的时间
			400 kHz 模式	1.3	—	ms	
			1 MHz 模式 (1)	TBD	—	ms	
D102	CB	总线容性负载		—	400	pF	

注 **1**： 对于所有 I²C™ 引脚，最小引脚电容为 10 pF。

2： 快速模式的 I²C 总线器件可用于标准模式的 I²C 总线系统中，但必须满足参数 #107 ≥ 250 ns 的要求。如果快速模式器件没有延长 SCL 信号的低电平周期，则必然满足此条件。如果该器件延长了 SCL 信号的低电平周期，它必须将下一个数据位输出到 SDA 线， SCL 线被释放前，在 100 kHz 模式下，参数 #102 + 参数 #107 = 1000 + 250 = 1250 ns。

图 26-19： USART 同步发送 （主控 / 从动）时序

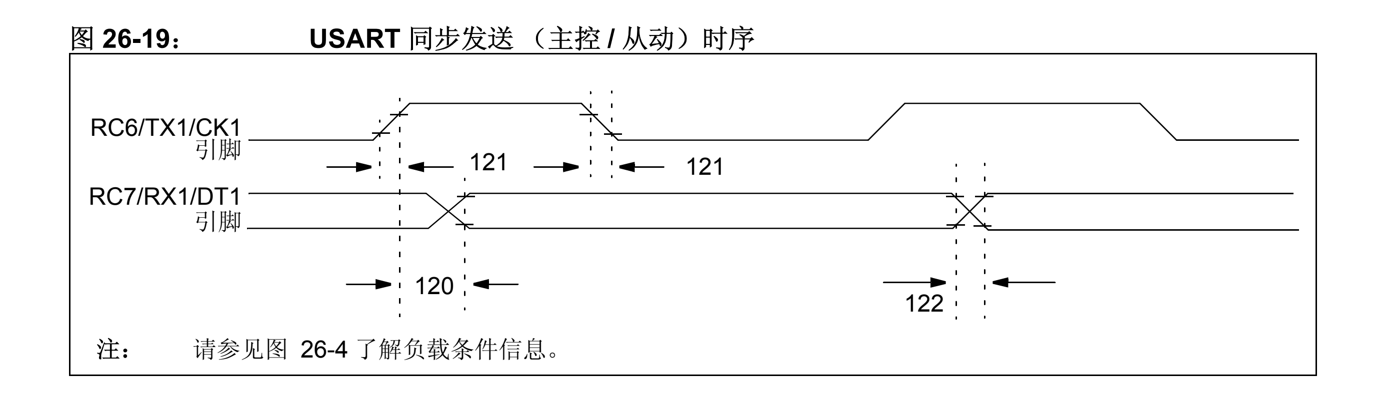
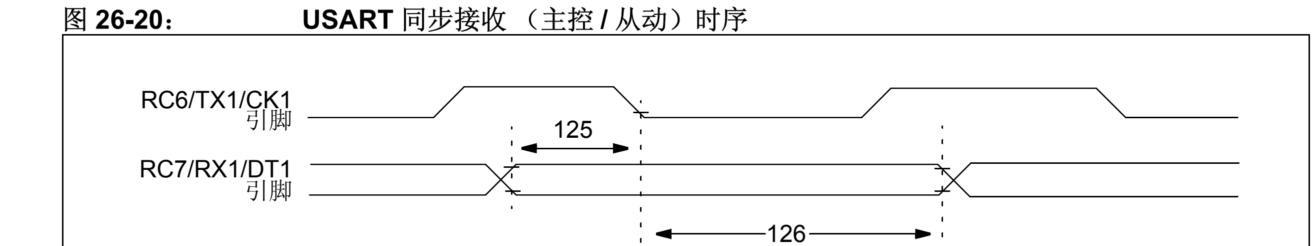


表 26-21： USART 同步发送要求

参数编号	符号	特性		最小值	最大值	单位	条件
120	TCKH2DTV	<u>SYNC XMIT （主控和从动模式）</u> 从时钟高电平至数据输出有效	PIC18**F**XXX	—	40	ns	
			PIC18**LF**XXX	—	100	ns	VDD = 2.0V
121	TCKRF	时钟输出信号的上升时间和下降时间（主控模式）	PIC18**F**XXX	—	20	ns	
			PIC18**LF**XXX	—	50	ns	VDD = 2.0V
122	TDTRF	数据输出信号的上升时间和下降时间	PIC18**F**XXX	—	20	ns	
			PIC18**LF**XXX	—	50	ns	VDD = 2.0V

图 26-20： USART 同步接收 （主控 / 从动）时序



注： 请参见图 26-4 了解负载条件。

表 26-22： USART 同步接收要求

参数编号	符号	特性	最小值	最大值	单位	条件
125	TDTV2CKL	<u>SYNC RCV （主控和从动模式）</u> 在 CK ↓之前数据的保持时间 （DT 保持时间）	10	—	ns	
126	TCKL2DTL	在 CK ↓之后数据的保持时间 （DT 保持时间）	15	—	ns	

表 **26-23**： **A/D** 转换器规范： **PIC18F2220/2320/4220/4320** （工业级）
PIC18LF2220/2320/4220/4320（工业级）

参数编号	符号	特性	最小值	典型值	最大值	单位	条件
A01	NR	分辨率	—	—	10	bit	ΔVREF ≥ 3.0V
A03	EIL	积分线性误差	—	—	<±1	LSb	ΔVREF ≥ 3.0V
A04	EDL	微分线性误差	—	—	<±1	LSb	ΔVREF ≥ 3.0V
A06	EOFF	偏移误差	—	—	<±1	LSb	ΔVREF ≥ 3.0V
A07	EGN	增益误差	—	—	<±1	LSb	ΔVREF ≥ 3.0V
A10	—	单调性	保证 (1)			—	
A20	DVREF	参考电压范围（VREFH – VREFL）	3	—	AVDD – AVSS	V	10 位精度时
A21	VREFH	参考电压高电平	AVSS+3.0V	—	AVDD + 0.3V	V	10 位精度时
A22	VREFL	参考电压低电平	AVSS – 0.3V	—	AVDD – 3.0V	V	10 位精度时
A25	VAIN	模拟输入电压	VREFL	—	VREFH	V	
A30	ZAIN	模拟电源阻抗的推荐值	—	—	2.5	kΩ	
A50	IREF	VREF 输入电流 （注 **2**）	— —	— —	±5 ±150	μA μA	在采集 VAIN 期间。 在 A/D 转换期间。

注 **1**： A/D 转换结果不会因输入电压的增加而减小，并且不会丢失代码。

2： VREFH 电流来自作为 VREFH 源的 RA3/AN3/VREF+/SEG1 引脚或 AVDD。
VREFL 电流来自作为 VREFL 源的 RA2/AN2/VREF-/SEG16 引脚或 AVSS。

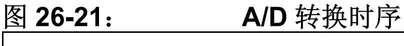
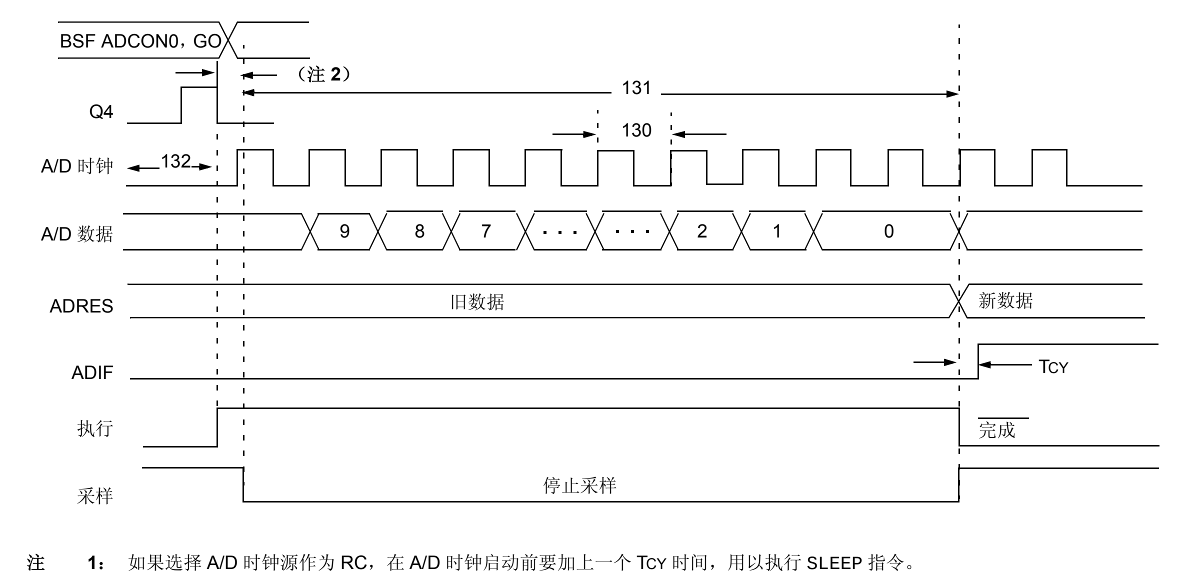


图 26-21： A/D 转换时序



注 1： 如果选择 A/D 时钟源作为 RC，在 A/D 时钟启动前要加上一个 TCY 时间，用以执行 SLEEP 指令。
2： 这是最小的 RC 延时 （典型值为 100 nS），在这段时间保持电容与模拟输入端是断开的。

表 26-24： A/D 转换要求

参数编号	符号	特性		最小值	最大值	单位	条件
130	TAD	A/D 时钟周期	PIC18**F**XXX	0.7	25.0[1]	μs	基于 OSC， VREF ≥3.0V
			PIC18**LF**XXX	1.4	25.0[1]	μs	VDD = 2.0V； 基于 TOSC，VREF 满量程
			PIC18**F**XXX	TBD	1	μs	A/D RC 模式
			PIC18**LF**XXX	TBD	3	μs	VDD = 2.0V； A/D RC 模式
131	TCNV	转换时间 (不包括采集时间) （注 2）		11	12	TAD	
132	TACQ	采集时间 （注 3）		1.4 TBD	— —	μs μs	-40°C 至 +85°C 0°C ≤ 至≤ +85°C
135	TSWC	转换→采样的切换时间		—	（注 4）		
TBD	TDIS	电容放电时间		0.2	—	μs	

图注： TBD = 待定

注 1： A/D 时钟周期的时间取决于器件频率和 TAD 时钟分频比。
2： 可在后续 TCY 周期内读 ADRES 寄存器。
3： 转换完成后当电压满量程变化时 （VDD 至 VSS，或 VSS 至 VDD），保持电容采样一个 “新”输入电压所需的时间。在输入通道上的电源阻抗 （*R*S）为 50Ω。
4： 在器件时钟的下一个周期上。

27.0 DC 和 AC 特性图表

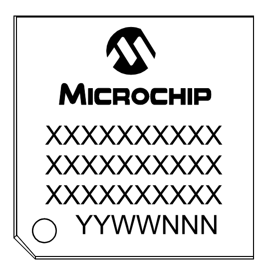
当前没有可用图表。

注：

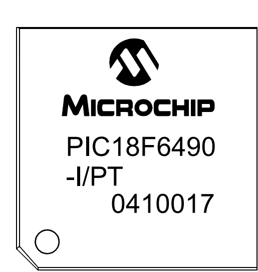
28.0 封装信息

28.1 封装标识信息

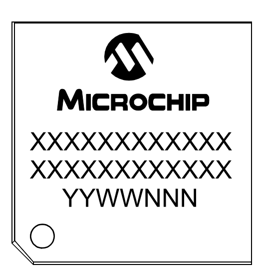
64 引脚 TQFP



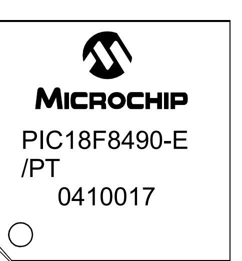
示例



80 引脚 TQFP



示例



图注：		
	XX...X	客户信息 *
	Y	年份代码（公历年的最后一位数字）
	YY	年份代码（公历年的最后两位数字）
	WW	星期代码（1 月 1 日的星期代码为 “01”）
	NNN	按字母数字排序的追踪代码

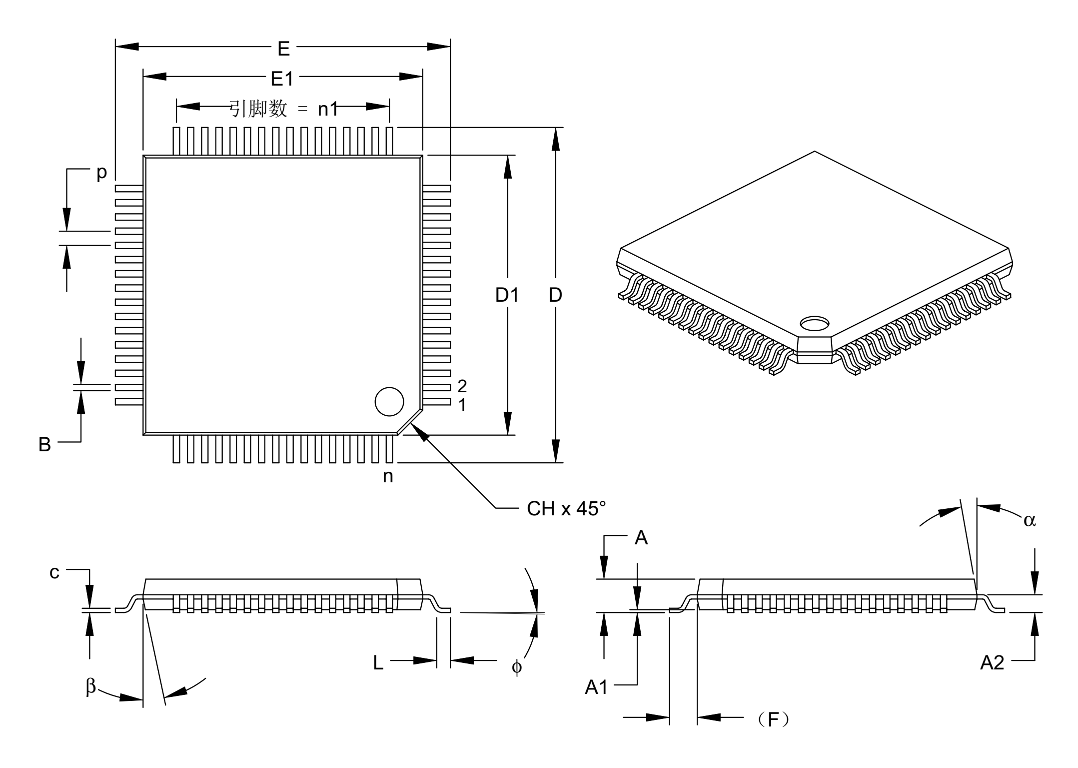
注： 若 Microchip 器件编号未在一行中完全标出，它将换行继续标出，因此会限制客户信息的可用字符数。

* 标准 PICmicro 器件标识包括 Microchip 器件编号、年份代码、星期代码和追踪代码。如需超过此范围的 PICmicro 器件标识，需支付一定的附加费用。请向当地的 Microchip 销售办事处确认相关信息。对于 QTP 器件，任何特殊标记的附加费用都已包含在 QTP 价格中。

28.2 封装详细信息

以下部分将介绍各种封装的技术细节。

64 引脚塑封薄型正方扁平封装 （PT）主体 10x10x1 mm， 1.0/0.10 mm 引脚形式 （TQFP）



单位		英寸			毫米*		
尺寸范围		最小值	正常值	最大值	最小值	正常值	最大值
引脚数	n		64			64	
引脚间距	p		.020			0.50	
每侧引脚数	n1		16			16	
总高度	A	.039	.043	.047	1.00	1.10	1.20
塑模封装厚度	A2	.037	.039	.041	0.95	1.00	1.05
悬空间隙	A1	.002	.006	.010	0.05	0.15	0.25
底脚长度	L	.018	.024	.030	0.45	0.60	0.75
引脚投影长度（参考）	(F)		.039			1.00	
底脚倾角	φ	0	3.5	7	0	3.5	7
总宽度	E	.463	.472	.482	11.75	12.00	12.25
总长度	D	.463	.472	.482	11.75	12.00	12.25
塑模封装宽度	E1	.390	.394	.398	9.90	10.00	10.10
塑模封装长度	D1	.390	.394	.398	9.90	10.00	10.10
引脚厚度	c	.005	.007	.009	0.13	0.18	0.23
引脚宽度	B	.007	.009	.011	0.17	0.22	0.27
引脚1切角斜面	CH	.025	.035	.045	0.64	0.89	1.14
塑模顶部锥度	α	5	10	15	5	10	15
塑模底部锥度	β	5	10	15	5	10	15

*控制参数

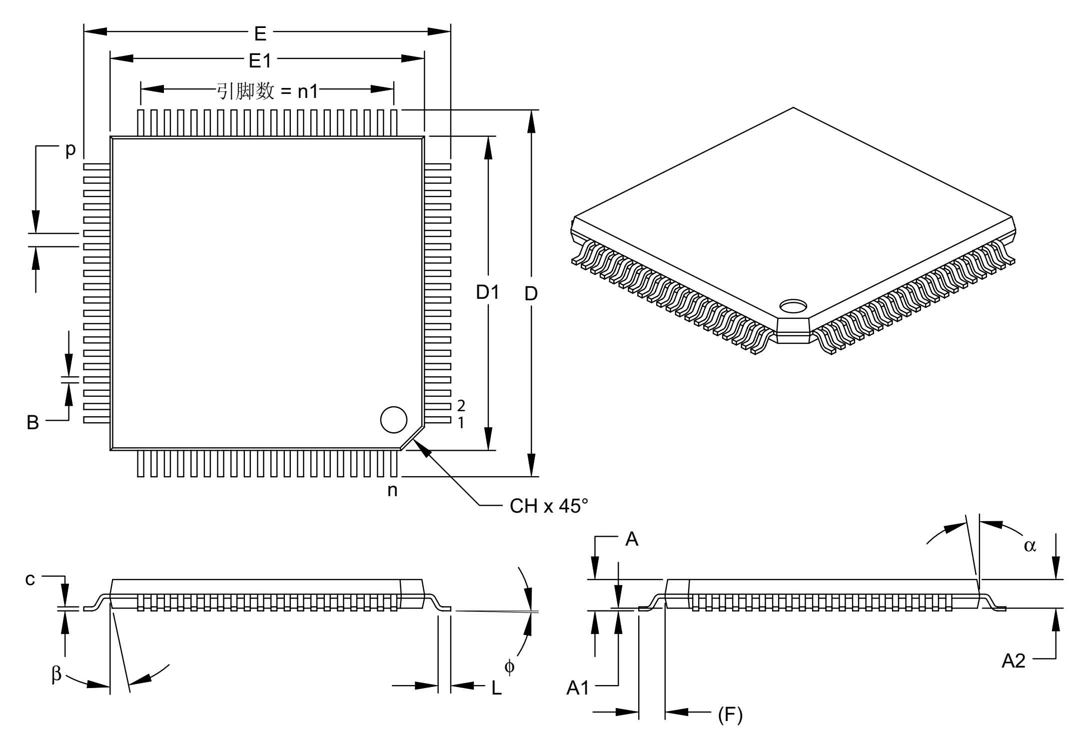
注：

尺寸D1和E1不包括塑模毛边或突起。塑模每侧的毛边或突起不得超过.010英寸（0.254mm）。

等同于JEDEC号：MS-026

图号：C04-085

80 引脚塑封薄型正方扁平封装 （PT）主体 12x12x1 mm， 1.0/0.10 mm 引脚形式 （TQFP）



单位		英寸			毫米*		
尺寸范围		最小值	正常值	最大值	最小值	正常值	最大值
引脚数	n		80			80	
引脚间距	p		.020			0.50	
每侧引脚数	n1		20			20	
总高度	A	.039	.043	.047	1.00	1.10	1.20
塑模封装厚度	A2	.037	.039	.041	0.95	1.00	1.05
悬空间隙	A1	.002	.004	.006	0.05	0.10	0.15
底角长度	L	.018	.024	.030	0.45	0.60	0.75
引脚投影长度	(F)		.039			1.00	
底角倾角	φ	0	3.5	7	0	3.5	7
总宽度	E	.541	.551	.561	13.75	14.00	14.25
总长度	D	.541	.551	.561	13.75	14.00	14.25
塑模封装宽度	E1	.463	.472	.482	11.75	12.00	12.25
塑模封装长度	D1	.463	.472	.482	11.75	12.00	12.25
引脚厚度	c	.004	.006	.008	0.09	0.15	0.20
引脚宽度	B	.007	.009	.011	0.17	0.22	0.27
引脚1切角斜面	CH	.025	.035	.045	0.64	0.89	1.14
塑模顶部锥度	α	5	10	15	5	10	15
塑模底部锥度	β	5	10	15	5	10	15

*控制参数

注：

尺寸D1和E1不包括塑模毛边或突起。塑模每侧的毛边或突起不得超过 .010英寸（0.254mm）。

等同于JEDEC号：MS-026

图号：C04-092

注：

附录 A： 版本历史

版本 A （2004 年 7 月）

PIC18F6390/6490/8390/8490 器件数据手册的最初版本。

附录 B： 器件差异

表 B-1 为本数据手册中所列器件之间的差异。

表 B-1： 器件差异

功能特性	PIC18F6390	PIC18F6490	PIC18F8390	PIC18F8490
LCD 驱动器可驱动的像素数量	128 （4 x 32）	128 （4 x 32）	192 （4 x 48）	192 （4 x 48）
I/O 端口	端口 A、 B、 C、 D、 E、 F 和 G	端口 A、 B、 C、 D、 E、 F 和 G	端口 A、 B、 C、 D、 E、 F、 G、 H 和 J	端口 A、 B、 C、 D、 E、 F、 G、 H 和 J
闪存程序存储器	8K 字节	16K 字节	8K 字节	16K 字节
封装类型	64 引脚 TQFP	64 引脚 TQFP	80 引脚 TQFP	80 引脚 TQFP

附录 C： 转换注意事项

本附录讨论了器件从老版本升级到本数据手册中所列版本时的注意事项。这些变化通常是由于采用的工艺技术之间的差异引起的，正如从 PIC16C74A 至 PIC16C74B 的升级。

不适用

附录 D： 从基本器件移植至增强型器件

本节讨论如何从基本器件（如 PIC16C5X）移植到增强型 MCU 器件 （如 PIC18FXXX）。

当前没有数据

附录 E： 从中档器件移植至增强型器件

在 AN716“Migrating Designs from PIC16C74A/74B to PIC18F442”中详细讨论了中档 MCU 器件（即 PIC16CXXX）与增强型器件（即 PIC18FXXX）之间的差异。虽然所讨论的内容都是针对特定器件的，但是适用于从中档器件移植至增强型器件的所有情况。

上述应用笔记的文献编号为 DS00716。

附录 F： 从高档器件移植至增强型器件

在 AN72 “PIC17CXXX to PIC18CXXX Migration”中详细讨论了从高档 MCU 器件（即 PIC16PIC17CXXX）移植至增强型器件（即 PIC18FXXX）的步骤以及这两类器件之间的差异。上述应用笔记的文献编号为 DS00726。

注：

索引

A

B

 初稿

G

H

J

K

L

 初稿

M

N

O

P

Q

R

S

 初稿

T

W

X

Y

Z

注：

MICROCHIP 网站

Microchip 网站（www.microchip.com）为客户提供在线支持。客户可通过该网站方便地获取文件和信息。只要使用常用的因特网浏览器即可访问。网站提供以下信息：

- **产品支持**——数据手册和勘误表、应用笔记和样本程序、设计资源、用户指南以及硬件支持文档、最新的软件版本以及存档软件
- **一般技术支持**——常见问题（FAQ）、技术支持请求、在线讨论组以及 Microchip 顾问计划成员名单
- **Microchip 业务**——产品选型和订购指南、最新 Microchip 新闻稿、研讨会和活动安排表、Microchip 销售办事处、代理商以及工厂代表列表

变更通知客户服务

Microchip 的变更通知客户服务有助于客户了解 Microchip 产品的最新信息。注册客户可在他们感兴趣的某个产品系列或开发工具发生变更、更新、发布新版本或勘误表时，收到电子邮件通知。

欲注册，请登录 Microchip 网站 www.microchip.com，点击“变更通知客户（Customer Change Notification）”服务后按照注册说明完成注册。

客户支持

Microchip 产品的用户可通过以下渠道获得帮助：

- 代理商或代表
- 当地销售办事处
- 应用工程师（FAE）
- 技术支持
- 开发系统信息热线

客户应联系其代理商、代表或应用工程师（FAE）寻求支持。当地销售办事处也可为客户提供帮助。本文档后附有销售办事处的联系方式。

也可通过 http：//support.microchip.com 获得网上技术支持。

读者反馈表

我们努力为您提供最佳文档，以确保您能够成功使用 Microchip 产品。如果您对文档的组织、条理性、主题及其他有助于提高文档质量的方面有任何意见或建议，请填写本反馈表并传真给我公司 TRC 经理，传真号码为 86-21-5407-5066。

请填写以下信息，并从下面各方面提出您对本文档的意见。

致：　TRC 经理　　　　　　　　　　总页数 ________

关于：　读者反馈

发自：　姓名 ________________________________

　　　　公司 ________________________________

　　　　地址 ________________________________

　　　　国家 / 省份 / 城市 / 邮编 ________________________________

　　　　电话：（______）________________　　　传真：（______）________________

应用（选填）：

您希望收到回复吗？是____　否____

器件：　PIC18F6390/6490/8390/　　文献编号：　DS39629B_CN

问题

1. 本文档中哪些部分最有特色？

____

____

2. 本文档是否满足了您的软硬件开发要求？如何满足的？

____

____

3. 您认为本文档的组织结构便于理解吗？如果不便于理解，那么问题何在？

____

____

4. 您认为本文档应该添加哪些内容以改善其结构和主题？

____

____

5. 您认为本文档中可以删减哪些内容，而又不会影响整体使用效果？

____

____

6. 本文档中是否存在错误或误导信息？如果存在，请指出是什么信息及其具体页数。

____

____

7. 您认为本文档还有哪些方面有待改进？

____

____

PIC18F6390/6490/8390/8490 产品标识体系

欲订货，或获取价格、交货等信息，请与我公司生产厂或销售办事处联系。

器件编号	X	/XX	XXX
器件	温度范围	封装	编程模式

器件	PIC18F6390/6490/8390/8490[1] 和 PIC18F6390/6490/8390/8490T[2] VDD 范围为 4.2V 至 5.5V PIC18LF6390/6490/8390/8490[1] 和 PIC18LF6390/6490/8390/8490T[2] VDD 范围为 2.0V 至 5.5V
温度范围	I = -40°C 至 +85°C（工业级） E = -40°C 至 +125°C（扩展级）
封装	PT = TQFP （薄型正方扁平封装）
编程模式	QTP、 SQTP、代码或特殊要求 （其他情况空白）

示例：

a) PIC18LF6490-I/PT 301 表示工业级温度、TQFP 封装、扩展级 VDD 范围和 QTP 模式 #301。

b) PIC18F8490-I/PT 表示工业级温度、 TQFP 封装、普通 VDD 范围。

c) PIC18F8490-E/PT 表示扩展级温度、 TQFP 封装、普通 VDD 范围。

注 1： F = 标准电压范围
LF = 宽电压范围
2： T = 卷带式封装